MUSIC BEYOND AIRPORTS

APPRAISING AMBIENT MUSIC

Edited by Monty Adkins & Simon Cummings

10.5920/beyondairports.fulltext

Published by University of Huddersfield Press

University of Huddersfield Press
The University of Huddersfield
Queensgate
Huddersfield HD1 3DH

Email enquiries university.press@hud.ac.uk

First published 2019

A CIP catalogue record for this book is available from the British Library.

ISBN: 978-1-86218-161-8

Designed by Dawn Cockcroft

10.5920/beyondairports.fulltext

CONTENTS

10.5920/beyondairports.fulltext

10.5920/beyondairports.fulltext

9

ADAPTIVE GAME SCORING WITH AMBIENT MUSIC

Axel Berndt

Game scoring: music meets interaction

Music is an integral part of video games. Since the minimalistic beeps and bleeps of the earliest games the overwhelming majority come with their own more or less elaborate music. Similar to early film music,[1] early game music served as a replacement for missing sound effects, which was still the case in many video games of the 1990s.[2] For several decades, gaming hardware was hardly able to produce naturalistic sound effects due to technological limitations. Sound synthesis technology for games of this time included standard waveform oscillators (such as square, saw and triangle), limited amplitude and frequency modulation, and basic filters (offered by only few sound chips). The stylized sound effects they produce established a unique aesthetic that continues today as part of our media culture, for instance in the chiptune genre. The uniqueness of these aesthetics becomes salient when experienced in other media contexts where naturalistic sounds are more common, such as in films, e.g. in *The Super Mario Bros. Super Show!*,[3] an animated television show that is based on the eponymous Nintendo games series.[4] These sound effects are closer to musical idents than filmic Foley aesthetics and can still be found today, for instance when games reward

1 Zofia Lissa, *Ästhetik der Filmmusik*. Henschel (Leipzig, Germany, 1965).

2 Nils Dittbrenner, "Soundchip-Musik: Computer- und Videospielmusik von 1977-1994," *Beiträge zur Medienästhetik der Musik* vol. 9, (University of Osnabrück, Osnabrück: epOs-Music, 2007).

3 DIC Entertainment. *The Super Mario Bros. Super Show!* Viacom, Nintendo, DHX Media, Sept. 1989. animated TV show.

4 Nintendo EAD. *Super Mario Bros.* Nintendo, Sept. 1985. music by Koji Kondo.

player achievements. They are often designed such that they integrate well into the background music. The result can sometimes establish a form of live-generated musical collage that reflects the events in the interactive scene.

Such a direct relation between interaction and music on the level of individual sound events (e.g. notes and drum kicks) is a typical feature of music video games,[5] such as *Rez*.[6] In the context of video games in general, music video games are a very specific genre regarding the integration of music as a core element of the games' mechanics. The link between interaction and music is usually established on a more abstract level, often with whole pieces of music that correspond with locations in the diegesis (the virtual world) or states of the narration. Nonetheless, in any case music has to follow the interactive progress of the game and serve various purposes.

Most of the narrative functions of music in video games are directly linked to the heritage of music in films. The music mediates emotions, associations, and descriptions of the setting or of physical activity (e.g., Mickey Mousing), leads the audiences' attention, and affects their sense of time.[7] Music does this either in parallel with the dramaturgy, complementary to it, or expresses a contrary/contrapuntal semantic level.[8] In video games these narrative concepts have to be reconsidered in the context of interaction. While acting within the diegesis, the player also perceives non-diegetic music, i.e. music that plays outside the virtual scene (in the 'off', just like a narrator's voice that is audible only to the audience). A sudden cue of battle music could

5 Axel Berndt, "Diegetic Music: New Interactive Experiences," *Game Sound Technology and Player Interaction: Concepts and Developments*, ed. Mark Grimshaw, (Hershey, PA.: IGI Global, 2011), 60-76.

6 Sega. *Rez*. Sega, 2001.

7 Johnny Wingstedt. *Making Music Mean: On Functions of, and Knowledge about, Narrative Music in Multimedia*, Unpublished PhD thesis (Luleå University of Technology, Department of Music and Media, Luleå, Sweden, August 2008).

8 See Sergej M. Eisenstein, Wsewolod I. Pudowkin, and Grigorij W. Alexandrow, "Manifest zum Ton-Film," (1928) *Texte zur Theorie des Films*, ed. Franz-Josef Albersmeier (Germany: Reclam, 1998), 3rd edition; and Wolfgang Thiel, *Filmmusik in Geschichte und Gegenwart* (Berlin: Henschel Verlag, 1981).

 10.5920/beyondairports.fulltext

now warn the player of an upcoming danger that was not yet discernible.[9] Originally non-diegetic music can influence the diegesis through its effect on the player. This situation is unknown to the film medium and unique to interactive media. It motivates Kristine Jørgensen's introduction of the conceptual layer of "trans-diegesis"[10] to video game theory.

This establishes a quasi-dialogue between music and player and leads to new narrative functions of music. It can support the player in playing the game by mediating pacing and quality of upcoming events. The music stages in the platform game *Rayman Legends*[11] are prime examples of this concept. In music video games it even dictates interactions, e.g. in the *Guitar Hero* series.[12] The associative power of music can be utilised to influence decision-making processes. Its kinaestetic energy affects playing behavior and dynamics (e.g., forward-driving or soothing), an effect that is also utilised in movement therapy and endurance training.[13] By evoking certain emotionality in response to player interaction, music can affect the player's self-conception and motivate irony or critical reflection of his or her decisions and actions. Music may also help to clarify consequences that would otherwise be unclear in a complex game through its mechanics and its mere visual presentation.

---

9 Axel Berndt, "Im Dialog mit Musik: Zur Rolle der Musik in Computerspielen," *Kieler Beiträge zur Filmmusikforschung*, vol. 9, (March 2013), 293-323.

10 Kristine Jørgensen, "Time for New Terminology? Diegetic and Non-Diegetic Sounds in Computer Games Revisited," *Game Sound Technology and Player Interaction: Concepts and Developments*, ed. Mark Grimshaw (Bolton: IGI Global, 2010).

11 Ubisoft Montpellier. *Rayman Legends*. Ubisoft August 2013.

12 Harmonix. *Guitar Hero* series [Computer games]. Harmonix, Neversoft, Vicarious Visions, Budcat Creations, FreeStyleGames (Developers), RedOctane, Activision (publishers), 2006-2017.

13 See Iris Bräuninger and Elisabeth Blumer, "Tanz- und Bewegungstherapie," *Psychiatrische Rehabilitation*, ed. Wulf Rössler, (Berlin: Springer, Berlin, 2004), 380-387; and Julian Rubisch, Matthias Husinsky, Jakob Doppler, Hannes Raffaseder, Brian Horsak, Beate Ambichl, and Astrid Figl, "A mobile music concept as support for achieving target heart rate in preventive and recreational endurance training," *Audio Mostly 2010: 5th Conference on Interaction with Sound—Sound and Design* (Piteå, Sweden: Interactive Institute/Sonic Studio Piteå, ACM, 2010), 142-145.

Most of these effects work best when the player perceives the music subconsciously.[14] This leads to an often misunderstood claim: background music shall not be heard.[15] This, however, does not describe an acoustic quality but refers to the mode of subconscious perception. It can be achieved by an interplay of several musical characteristics such as a low density of musical events, soft amplitude envelopes of the sounds and a less memorable formal structure. Subtle cues and endings are implemented by slowly fading, starting and ending on very high or low pitches. Cues can be masked by louder sound effects. As such, the aesthetics of the ambient music genre seem custom-fit for this subconscious listening mode. Brian Eno writes:

> Ambient Music must be able to accommodate many levels of listening attention without enforcing one in particular: it must be as ignorable as it is interesting.[16]

Throughout the remainder of this section we will follow this concept further. Ambient music comes with several advantages that prove its worth for adaptive game scoring and explains its frequent occurrence in video games throughout the past decades. Hence, ambient aesthetics will also be the basis of our explorations of generative game music techniques later in this text.

Beyond the perceptual attitude, interaction comes with two further core challenges for musical accompaniment.

1. The progress of an interactive scene and its particular timing are unpredictable. How long does a situation last? Music has to wait for that same period. Endlessly repeating a piece of music might be the

14 Norbert J. Schneider, *Handbuch Filmmusik I: Musikdramaturgie im neuen Deutschen Film* (Munich: Verlag Ölschläger, 1990) 2nd Edition.

15 Theodor W. Adorno and Hanns Eisler, *Composing for the Films* (New York: Oxford University Press, 1947).

16 Bill Milkowski, "Brian Eno: Excursions in the Electronic Environment," *Down Beat: The Contemporary Music Magazine*, (June issue), 1983.

 10.5920/beyondairports.fulltext

easiest and most common workaround if the scene is longer than the piece. But, once the player becomes aware of it, it produces a déjà vu-like repetitiveness that usually contradicts the progress of the interactive scene. This can become frustrating and even annoying. "The user may tire of the same musical accompaniment and consider it monotonous or even irritating".[17]

2. Players do not wait for the music. Music has to follow. A certain musical change can be required at any time, long before the currently playing piece is finished. This situation is typically resolved by hard cuts (skip the current piece and start the new one) and soft cuts (quickly fade out the current piece and start the new one). Both are asynchronous to the music, do not even comply with basic musical aesthetics (such as its rhythmic properties, hence, have to be judged unmusical and disruptive.

Generally speaking, linear pieces of music are not as well-suited to follow developments of a nonlinear, interactive narration in a musically coherent fashion.

Thanks to its compositional paradigms, ambient music can maintain a certain atmosphere or character of expression for a surprisingly long time without interest waning. This makes it an ideal model to approach the first challenge. Eno's most famous track "1/1" from his album *Ambient 1: Music for Airports*[18] has a running time of 17 minutes and 21 seconds. The *Skyrim Atmospheres*[19] (part of the *The Elder Scrolls V: Skyrim* soundtrack album) has a running time of 42 minutes and 34 seconds. Other ambient tracks can play for several hours. The underlying paradigms of ambient music are characterized by Thom Holmes as follows:

---

17 Todor C. Fay, *A System and Process for Composing Musical Sections*. U.S. Patent No. 5,753,843. May, 1998.

18 Brian Eno, *Ambient 1: Music for Airports* (vinyl) (UK, Polydor Records – AMB001, 1978).

19 Jeremy Soule, *The Elder Scrolls V: Skyrim Original Game Soundtrack* (Bethesda Softworks, 2011).

> If there is a unifying element in all ambient music it appears to be a continuity of energy that enables a suspension of tension. Like minimalism, contemporary ambient music often relies on a persistent rhythm and slowly evolving wash of sound textures.[20]

The "establishment and maintenance of a single pervasive atmosphere" is achieved by "non-developmental forms, regularly or irregularly repeating events or cycles of events, modal pitch-sets, choice of a few limited parameters for each piece, and a pulse that is sometimes uneven, sometimes 'breathing', and sometimes non-existent" as Eric Tamm[21] summarizes his analysis of Eno's ambient music. Many ambient compositions do not even rely on repetition. They constantly vary and wander around, but once a mode of expression is established, it is maintained. These qualities are the essential prerequisites to comply with the need to wait for the player in a video game for an unknown period.

The impression of timelessness in many ambient compositions is reinforced by sonic properties. Three major categories of sounds can be identified.

1. Sounds with bell-like amplitude envelopes such as piano, vibraphone, harp, and bells, (i.e. with a percussive attack, effectively no sustain and a long release phase) seem to fade away endlessly. These often arpeggiate more or less randomly over an underlying pitch-set, chord or scale;
2. Pad and drone sounds such as choir, strings, and synth pads, with smooth attack and release phases serve mostly as chordal instruments and sustained sounds;
3. Atmospheric sounds (nature, urban, synthetic) may occur less frequently in autonomous ambient music productions. In the video

20 Thom Holmes, *Electronic and Experimental Music: Technology, Music, and Culture* (New York: Routledge, 2012), 4th edition.

21 Eric Tamm, *Brian Eno: His Music and the Vertical Color Of Sound* (Boston, MA.: Da Capo Press, 1995).

 10.5920/beyondairports.fulltext

> game context, however, they are a regular element that originates from the virtual scene's soundscape, as can be heard in the *Skyrim Atmospheres*. It seems natural to incorporate these sounds to some degree as part of the musical conception.

Monophonic (lead) melodies, such as in Brian Eno and Harold Budd's "Not Yet Remembered"[22], are rare. Most ambient compositions avoid easily recognizable elements and structures for which a melody would be a prime example. Amelodic and often arhythmic sequences of overlapping tones are used instead. Phrases of increasing and decreasing density of musical events resemble slow and deep breathing. Rests of up to 10 seconds can be observed between such phrases. These are not completely silent, though, but filled with the tone releases and long reverberations.

On the compositional level, ambient music is often constructed in a serial or procedural fashion. The whole range of generative music approaches is applicable, such as layered event loops, variation through permutation, fractal structures, and Monte Carlo methods.[23] This, ultimately, paves the way for the claim of short-term reactivity of video game music (the second challenge). The following section provides an introduction to, and classification of, approaches to incorporate nonlinearity in musical playback.

A taxonomy of adaptive music techniques

Music in video games - and in interactive media in general - must be capable of waiting for an unpredictable amount of time while at the same time reacting at short notice when player interaction triggers progress. These opposing claims are commonly accomplished by arrangement techniques in two musical dimensions: sequential and parallel. This section presents a taxonomy of these and further, more elaborate approaches, categorized by

22 Brian Eno and Harold Budd. *Ambient 2: The Plateaux of Mirror* (UK: Editions E.G. – EGAMB002, 1980).

23 Gerhard Nierhaus. *Algorithmic Composition: Paradigms of Automated Music Generation*. (Vienna: Springer Verlag, 2009).

their (increasing) use of generative techniques. It discusses the gain of musical flexibility that they exploit. Figure 1 gives an overview of this taxonomy.

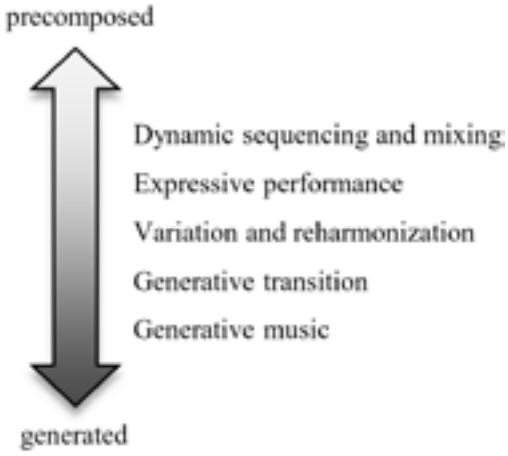

Fig. 1: A taxonomy of approaches to introduce nonlinearity to video game music. The axis indicates the reliance on precomposed and generated musical material.

The concept of sequential arrangement or *dynamic sequencing* reorganizes a sequence of musical segments in real-time according to the interactive context. Its roots lie in classical musical dice games.[24] An example of this technique applied to video games can be found in *Ghost Master*.[25] One of the most well-known implementations of the concept is the *iMuse* interactive music engine[26] that has been used in several LucasArts video games such as

24 See Johann Philipp Kirnberger, *Der allezeit fertige Polonaisen und Menuetten Komponist* (Berlin: George Ludewig Winter, 1767); Wolfgang Amadeus Mozart, *Musikalisches Würfelspiel: Anleitung so viel Walzer oder Schleifer mit zwei Würfeln zu componieren ohne musikalisch zu seyn noch von der Composition etwas zu verstehen* (Köchel Catalog of Mozart's Work KV1 Appendix 294d or KV6 516f, 1787); and Scott Joplin, *Melody Dicer* (New York: Carousel Publications, 1974).

25 Sick Puppies. *Ghost Master*. Empire Interactive, May 2003. Music by Paul Weir.

26 Michael Land and Peter N. McConnell, *Method and apparatus for dynamically composing music and sound effects using a computer entertainment system* (United States Patent No. 5,315,057, May 1994, filed November 1991).

 10.5920/beyondairports.fulltext

Monkey Island 2[27] and *X-Wing*.[28] Instead of loose musical segments, *iMuse* uses whole pieces, segmented by jump marks which can be activated to redirect the playback to other pieces and prefabricated transitions in-between. Since the playback of the current segment will not be interrupted, the reactivity of the music depends on the length of its segments. New musical material can only come up with the next segment. Long segments reduce reactivity while short segments tend to make the music 'short of breath' and prevent bigger formal structures such as overarching melodic lines.

Parallel arrangement applies *dynamic mixing* of a multitrack recording. Different combinations of audio tracks are transitioned via fade-ins and fade-outs and present different musical material. A typical example would be: a driving percussion track being added when an action scene starts and fading out when the action is over. The concept of *parallel composing* is a subspecies of dynamic mixing where the playback jumps dynamically between parallel audio tracks.[29] With dynamic mixing the overall playback and musical flow remain unaffected. Fading can be done at any time with no latency which makes the music very reactive while staying coherent. However, to ensure this coherency all parallel tracks have to harmonize and follow the same meter to ensure synchronicity. This limits the amount of change that parallel arrangement offers, i.e. it cannot transition to something fundamentally different. Approaches that combine both

27 LucasArts, *Monkey Island 2: LeChuck's Revenge*, 1991. Music by Michael Land, Peter McConnell, and Clint Bajakian.

28 LucasArts, *Star Wars: X-Wing*, 1993. Music by Michael Land, Peter McConnell, and Clint Bajakian.

29 Tim van Geelen, "Realizing ground-breaking adaptive music," *From Pac-Man to Pop Music: Interactive Audio in Games and New Media*, ed. Karen Collins (Aldershot: Ashgate, 2008), 94-102.

dynamic sequencing and mixing, are capable of overcoming this limitation.[30] Two video games that utilize dynamic mixing as a core method for their musical nonlinearity are Rockstar Games' (2010) *Red Dead Redemption*[31] and Strange Loop Games' (2012) *Vessel*.[32]

Music in today's games is dominated by pre-recorded audio material (often orchestral but also electronic music) that is specifically composed and prepared during the production process to enable the described arrangement techniques. Further, more flexible approaches to nonlinearity require a representation of music that provides more differentiated access and editing. The MIDI standard, for instance, gives access to every single note event allowing for changes of pitch, velocity and duration split-seconds before it is played back by a synthesiser. A synthesiser's parameters can be modulated in real-time by MIDI control and OSC (Open Sound Control) messages. Some Amiga and MS-DOS games used trackers, special types of musical sequencers, for their music which offered the same kind of freedom and editability during playback. Trackers are still in active use today, especially in the chiptune and demoscene. New tracker software releases from recent years (such as SVArTracker, Aodix, Hackey-Trackey, and reViSiT) also integrate the VST standard, run on modern operating systems and feature modern graphical user interfaces - evidences of this technology's up-to-dateness that is also applicable to modern games. This openness paves the way to introduce more flexible concepts to music beyond the possibilities that pre-recorded audio can practically offer.

---

30 See Sebastian Aav. *Adaptive Music System for DirectSound*. Unpublished Master's thesis (University of Linköping, Department of Science and Technology, Norrköping, Sweden, December 2005); Axel Berndt, Knut Hartmann, Niklas Röber, and Maic Masuch, "Composition and Arrangement Techniques for Music in Interactive Immersive Environments" *Audio Mostly 2006: A Conference on Sound in Games* (Piteå, Sweden: Interactive Institute/Sonic Studio Piteå, ACM, 2010), 53-59; and Harald Tobler. *CRML - Implementierung eines adaptiven Audiosystems* (Master's thesis, Fachhochschule Hagenberg, Medientechnik und design, Hagenberg, Austria, July 2004).

31 Rockstar Games. *Red Dead Redemption*. Rockstar San Diego, May 2010.

32 Strange Loop Games. *Vessel*. indiePub, March 2012. Music by Jon Hopkins, adaptive music system by Leonard J. Paul.

 10.5920/beyondairports.fulltext

A rarely addressed domain for musical nonlinearity is the real-time variation of its expressive performance. Tempo, dynamics and articulation coin its expressive character.[33] Berndt[34] presents computational models to emulate human musicians' performances. These can be used in combination with a mark-up or scripting language such as MPM (Music Performance Markup[35]) to define and render different performances of a piece of music. On this basis, seamless transitions can be computed during performance by simulating the reactions of virtual musicians.[36] The musical material can still be pre-composed which yields a relatively easy integration with established workflows of composers and game music studios. However, the bandwidth of expressions is rather limited by that musical material. The performance can bend it to a certain extent but the compositional substance (melodies, motifs, chords, rhythms etc.) stays the same.[37] Hence, this approach is best suited to short-range narrative processes that take place within a homogeneous scene or section of the game, for example a puzzle that has to be solved in several steps, a platform game where music reflects the progress within a stage, or a multiplayer match that features several phases of gameplay until it is won (e.g. attack, trace, defend).

Considering the limitations of the expressive performance approach, another idea comes to mind immediately: real-time variations of a given

33 Steven R. Livingstone, Ralf Muhlberger, Andrew R. Brown, and William F. Thompson, "Changing Musical Emotion: A Compositional Rule System for Modifying Score and Performance," *Computer Music Journal*. Vol. 34, no. 1, (Cambridge, MA.: The MIT Press, 2010), 41-64.

34 Axel Berndt, "Formalizing Expressive Music Performance Phenomena," *Works in Audio and Music Technology*, ed. Axel Berndt, (Dresden: TUDpress, 2015), 97-128.

35 Axel Berndt and Benjamin W. Bohl, "Music Performance Markup: Formale Beschreibung musikalischer Interpretationen," *International Yearbook of Scholarly Editing*, eds. Rüdiger Nutt-Kofoth and Bodo Plachta, Vol. 32, (Berlin: De Gruyter, 2018) 185-204.

36 Axel Berndt, "Decentralizing Music, Its Performance, and Processing," *Proceedings of the International Computer Music Conference (ICMC)*, eds. Margaret Schedel and Daniel Weymoth, (Stony Brook University, New York: International Computer Music Association (ICMA), 2010), 381-388.

37 The combination of nonlinear expressive performance and dynamic mixing can further increase the bandwidth of expressions beyond the limits of each individual technique.

music on a compositional level. This may start with changing the underlying scale.[38] Transforming a piece from, for example, C major to C harmonic minor can be achieved by replacing each E by an E^b, and each A by an A^b. This tonality variation alone can change the character of expression, especially when combined with a change in its performance. It is further possible to change the whole harmonization by adapting the voice leadings in the polyphonic counterpoint so that different chords are played. A planning approach for effective reharmonizations based on a measure for chord tension is given by Min-Joon Yoo and In-Kwon Lee.[39] Michael Stenzel[40] describes an algorithmic approach to reharmonization. Replacing a chord at a certain position in the music is achieved by shifting every note at that position that does not fit into the new chord to the nearest fitting pitch. For instance, a change from a C major chord to an A minor chord is done by shifting every G two semitones up to A. In real-life situations, though, reharmonization is not as easy as in this simplified example, as Stenzel's experiments revealed. Non-chord tones require a different treatment, differing chord complexities (e.g. a change from a triad to a five-note chord) complicate the mapping, the changed chord progressions may contradict melodic gestures, and adaptations of voice leadings can easily conflict with melodic/motivic structures. Thus, a fully satisfying algorithm for reharmonization must still be considered an open problem.

Real-time algorithmic variation of a given musical piece is not limited to reharmonization, which appears to be relatively rigid due to its various constraints. Further flexibility can be exploited by melodic variation, for example adding ornamentations to a melodic line or replacing it by an

38 Steven R. Livingstone et al., "Changing Musical Emotion."

39 Min-Joon Yoo and In-Kwon Lee, "Musical Tension Curves and its Applications," *Proceedings of the International Computer Music Conference (ICMC)* (New Orleans, Louisiana: International Computer Music Association (ICMA), 2006), 482-486.

40 Michael Stenzel, *Automatische Arrangiertechniken für affektive Sound-Engines von Computerspielen* (Diploma thesis, Otto-von-Guericke University, Magdeburg, Germany, 2005).

 10.5920/beyondairports.fulltext

alternative sequence of notes that may be derived from the original, or a free improvisation that introduces new melodic material. Both can be generated algorithmically and on the fly during playback. Melody embellishment is subject to music theory already for several centuries,[41] mostly in form of catalogue-like collections of ornaments. Notated examples from the Baroque era can be found in Georg Philipp Telemann's *12 Methodische Sonaten*.[42] On a note-by-note level, simple melodic embellishment is algorithmically relatively easy to achieve by adding transitioning figures such as non-chord notes from classical counterpoint.[43] These can be further embellished in a fractal fashion to create more elaborate ornaments. Alexander Putman and Robert Keller[44] developed a transformational grammar that implements jazz idioms to generate solo improvisations. Its technical implementation is part of the Impro-Visor[45] software, an educational tool for jazz improvisation. Two artificial neural network-based systems that learn and generate melody variations, *MeloNet* and *JazzNet*, have been presented by Dominik Hörnel and Wolfram Menzel[46] and were demonstrated using the styles of Johann Pachelbel and Charlie Parker. A very successful system for melodic

41 See Günter Altmann, *Musikalische Formenlehre—Ein Handbuch mit Beispielen und Analysen. Für Musiklehrer, Musikstudierende und musikinteressierte Laien* (Mainz: Schott, 2001), 8th revised edition; Joseph Müller-Blattau, ed., *Die Kompositionslehre Heinrich Schützens in der Fassung seines Schülers Christoph Bernhard* (Kassel: Bärenreiter, 1999). 3rd edition; Frederick Neumann, *Ornamentation in Baroque and Post-Baroque Music* (Princeton: Princeton University Press, 1978); and William Bloomfield Pepper, *The Alternate Embellishment in the Slow Movements of Telemann's "Methodical Sonatas"* (Unpublished PhD thesis, University of Iowa, Iowa City, Iowa, July 1973).

42 TWV 41: A3, a2, D3, E2, G4, g3, B5, C3, c3, d2, E5, h3.

43 Alfred Mann, *The Study of Counterpoint from Johann Joseph Fux's Gradus ad Parnassum* (New York: W. W. Norton & Company, Inc., 1971), revised edition.

44 Alexander M. Putman and Robert M. Keller, "A Transformational Grammar Framework for Improvisation," *Proceeding of the International Conference on New Music Concepts* (Treviso: March, 2015)

45 https://www.cs.hmc.edu/~keller/jazz/improvisor/, accessed April 9, 2018.

46 Dominik Hörnel and Wolfram Menzel, "Learning Musical Structure and Style with Neural Networks," *Computer Music Journal*, Vol. 22, no. 4 (Cambridge MA.: The MIT Press, 1999), 44-62.

improvisations is *GenJam*.[47] It runs an evolutionary algorithm to create call-and-response improvisations together with a co-performing human musician.

Reharmonization and variation of voice leadings (not only the melody) may also be utilized in the course of creating modulations and, thereby, seamless transitions to other pieces of music. The challenge is to adapt the currently playing piece so that the changes still conform to its aesthetics, i.e. appear to be pre-composed. As a first experiment in melody transitions, Max Mathews and Lawrence Rosler[48] applied linear interpolation methods using two melodies. The idea has been adopted by René Wooller and Andrew Brown[49] into one of their *Music Morphing* algorithms. Further algorithms are a weighted mixture of fragments from the start and target music using Markov morphing. The latter instantiates two Markov chains, one from the start music, the other from the target music, and applies them in an alternating fashion to generate the transition. While the start music Markov chain gets more time at the beginning of the transition, the ratio shifts in favour of the target music later on. A conceptually similar, evolutionary algorithm-based approach to combine two musical fragments into one is *MusicBlox*.[50] Its fitness function measures the generated candidates' similarity to both input pieces and triggers further recombination and mutation to achieve a desired similarity ratio. So far, these techniques have been applied as creative tools for

47 John Al Biles, "Improvizing with Genetic Algorithms: GenJam," *Evolutionary Computer Music*, eds. Eduardo R. Miranda and John Al Biles, (London: Springer, 2007), 137-169.

48 Max V. Mathews and Lawrence Rosler, "Graphical Language for the Scores of Computer-Generated Sounds," *Perspectives of New Music*, Vol. 6, no. 2 (1968), 92-118.

49 René Wooller and Andrew R. Brown, "Investigating morphing algorithms for generative music," *Third Iteration: Third International Conference on Generative Systems in the Electronic Arts* (Melbourne, Australia, 2005).

50 Andrew Gartland-Jones, "MusicBlox: A Real-Time Algorithmic Composition System Incorporating a Distributed Interactive Genetic Algorithm," *Proceedings of EvoWorkshop/EuroGP2003, 6th European Conference in Genetic Programming* (Berlin: Springer, 2003), 490-501.

 10.5920/beyondairports.fulltext

composition and live music making[51] but have barely been tested for transition generation in a gaming or similar context. It is likely that they work very well for certain genres and styles, such as serialism, minimalism and electronic dance music as these often apply similar compositional paradigms. Mathews and Rosler,[52] however, after applying their interpolation approach to two folk melodies, judged the result "nauseating" and "jarring" - hence, this was a stylistic context that seems less compatible with these techniques. Besides these few academic approaches, generative real-time music transitioning is still an open problem.

So far, generative techniques have been introduced in order to perform, vary, adapt, and transition pre-composed music in real-time. The step to a fully generated game score is at hand and comes with a promising new perspective to the problem of nonlinearity. Instead of laborious editing of pre-composed pieces, a real-time music generation algorithm offers 'musical adjustment screws' in terms of its parameters to affect its musical output. These parameters can be modulated by the game engine, resulting in the music dynamically following the progress of the game.

A first instance of live generated game music can be found in the 1984 video game *Ballblazer*.[53] The title theme "Song of the Grid" is generated using Peter Langston's *riffology* technique.[54] A melody is built from a set of riffs (short melodic segments) using random choice and some rules for good melodic connections. Riffs can also be varied by changing the tempo and skipping notes.

---

51 Andrew R. Brown, René W. Wooller and Eduardo R. Miranda, "Interactive Evolutionary Morphing as a Music Composition Strategy," *Music As It Could Be: New Musical Worlds from Artificial Life*, ed. Eduardo R. Miranda (Madison: A&R Editions, 2009).

52 Max V. Mathews and Lawrence Rosler. "Graphical Language for the Scores of Computer-Generated Sounds."

53 David Levine and Peter S. Langston. *Ballblazer*. Lucasfilm Ltd., March 1984.

54 Peter S. Langston, "Six Techniques for Algorithmic Music Composition," *Proceedings of the International Computer Music Conference (ICMC)* (Columbus, Ohio: Ohio State University, 1989), 164-167. Extended version with code examples available at http://peterlangston.com/Papers/amc.pdf, accessed March 17, 2018.

> An endlessly varying melody is generated by riffology and played over an accompaniment consisting of a bass line, drum part, and chords. The accompaniment itself is assembled on the fly from a repertoire of four-bar segments, using a simplified version of the riffology technique.[55]

Langston's riffology technique can be regarded as an extension of the concept of dynamic sequencing but with segments of the length of a single riff. These are much shorter than the musically more sovereign segments that the above concept of dynamic sequencing involves. The riffs are conceptually closer to single notes and their arrangement requires a more compositional logic such as riffology to form a musical idea from them.

The situation is similar with the music of Hello Games' *No Man's Sky*.[56] Its generative music system PULSE integrates into the Wwise audio engine.[57] According to Paul Weir, audio director of *No Man's Sky*, "Pulse, at its heart, is really just a glorified random file player".[58] An instrument is defined by a collection of sounds, which can go down to single tones or notes - so basically a form of sampling. The audio material is a decomposition of original pieces from the band 65daysofstatic. One such instrument holds "variations of a single type of sound" and is associated with a certain "playback logic, such as how often the sound can play, its pitch, pan and volume information".[59] The music system is capable of interpolating between states, thus, creating seamless transitions between different musical soundscapes. In this

55 Ibid.

56 Hello Games, *No Man's Sky*. Aug. 2016. Music composed by 65daysofstatic, music system and logic by Paul Weir.

57 Audiokinetic Inc. *Wwise (Wave Works Interactive Sound Engine)*, https://www.audiokinetic.com/products/wwise/, access: May 7, 2018.

58 Anne-Sophie Mongeau, "Behind the Sound of 'No Man's Sky': A Q&A with Paul Weir on Procedural Audio," *A Sound Effect*. March 2017, https://www.asoundeffect.com/no-mans-sky-sound-procedural-audio/, accessed May 8, 2018.

59 Ibid.

 10.5920/beyondairports.fulltext

conceptual framework, 65daysofstatic's original compositions can be regarded as blueprints for the music scripting. The scripts, however, do not merely reproduce them but allow for more variation.

Throughout recent years, game audio engines have become more versatile in terms of real-time sound synthesis and digital signal processing (DSP) capabilities as well as programmability. This facilitates the integration of more advanced procedural music techniques. Unreal Engine 4[60] introduced Blueprints Visual Scripting[61] as a replacement for the former Unreal Script. It is a graphical scripting language similar to MAX and Pure Data. While primarily intended for gameplay scripting it also features some audio and music scripting features and can integrate custom C++ code that may add, for instance, DSP and algorithmic composition functionalities. Since Brinkmann et al.'s introduction of the *libPd* embeddable library,[62] Pure Data,[63] a widely used powerful tool for electronic musicians, has been frequently used in the gaming context on all major hardware platforms and operating systems. A mobile game that utilizes *libPd* for its generative music and real-time sound synthesis is *Sim Cell*.[64] Pure Data's front end is also used by the *heavy audio tools* framework[65] that allows for the creation of plugins for Unity 5, Wwise, Web Audio API and others. Hence, Pure Data became also a regular part of Leonard J. Paul's *School of Video Game Audio*.[66]

---

60 Epic Games, *Unreal Engine 4*. https://www.unrealengine.com, accessed May 8, 2018.

61 https://docs.unrealengine.com/en-us/Engine/Blueprints, accessed May 8, 2018.

62 Peter Brinkmann, Peter Kirn, Richard Lawler, Chris McCormick, Martin Roth, and Hans-Christoph Steiner, "Embedding Pure Data with libpd," *Fourth International Pure Data Convention 2011 – A Festival of Programmed Music and Art* (Berlin, August 2011).

63 Miller Puckette, "Pure Data," *Proceedings of the International Computer Music Conference (ICMC)* (Thessaloniki: International Computer Music Association (ICMA), 1997), 224-227.

64 Touch Press Games. *Sim Cell*. iOS app, v1.2.0, 2017. Music by Leonard J. Paul.

65 Enzien Audio Ltd. *heavy audio tools*. First release: January 2016. https://enzienaudio.com/, accessed May 9, 2018.

66 http://videogameaudio.com/, accessed May 9, 2018.

The developers of the video game *Spore*[67] created their own derivative of Pure Data for integration into their game engine, which is called *EApd*.[68] *Spore* incorporates both pre-composed pieces and procedural music. The development of the procedural music system was undertaken by Kent Jolly and Aaron McLeran with Brian Eno acting as consultant. Different scenes in the game (e.g. creature editor, UFO editor, civ editor) have their own Pure Data patches/algorithms. The game dynamically switches between them. Hence, nonlinear musical development basically unfolds within these scenes. Four main design goals led the music system's development:[69]

1. Make the music so that it is not distracting;
2. Music should be non-repetitive and along these lines: "music that is eminently unhummable";
3 Make it playful;
4. Music responds to the user.

Goals two and four are equivalent to the above mentioned two core challenges of music in video games. They reflect the conception of the music algorithm. Rhythmic and melodic patterns are created from random numbers. The seed values of the random number generators are stored and serve as hashes to recall the patterns - effectively motifs - at later occasions. The system can switch back to them by reseeding. Multiple loops of patterns are layered over each other and combined by random crossfades. Instrument switching, varying rhythmic density and reseeding are used to make the music sound more diverse. A melody generator is included that is based on Markov models. A further routine implements a simplified counterpoint based on

67 Maxis. *Spore*. Electronic Arts, September, 2008.

68 Kent Jolly, "Usage of Pd in Spore and Darkspore," *Fourth International Pure Data Convention 2011 – A Festival of Programmed Music and Art* (Berlin, August 2011).

69 Kent Jolly and Aaron McLeran, "Procedural Music in SPORE," *Game Developers Conference: GDC Vault*. 2008. https://www.gdcvault.com/play/323/Procedural-Music-in, accessed May 10, 2018.

 10.5920/beyondairports.fulltext

Johann Joseph Fux's treatise *Gradus ad Parnassum*.[70]

Real-time music generation and variation also requires real-time sound synthesis. Music production projects in sound studios often have several synthesisers and DSP effects processors running in parallel to achieve certain sound aesthetics. The required computing resources are far beyond what is typically available in gaming contexts. Leonard J. Paul[71] gives reference values: Audio in games usually gets 10% of CPU and 10% of memory. In comparison to studio production situations this can become quite a limiting factor. However, there is also much room for further optimisation. Many sounds do not have to be synthesised 'from scratch' but can be sampled. Once the procedural music is defined, its polyphony is also known and the sound generators (be it synthesisers or samplers) can be optimised accordingly by limiting their polyphony to what is actually necessary. Even the amount of sample data can be cut down significantly once the pitch range of each voice is known. At the end of the sound design and prototyping phase it is also a good case to incorporate a more or less extensive revision step in the workflow to clean up and replace cluttered, inefficient code by equivalent, more efficient alternatives. This may be more common in engineering than in artistic workflows, but it is worth the effort.

Experiments with generative ambient

Real-time generative music techniques offer various 'adjustment screws' to affect the music during playback and make it more flexible in order to adapt to an interactive context such as the narrative progress of a video game. The type and amount of flexibility will vary depending on the type of musical parameters to be edited and the aesthetic agenda that the music has to comply with. The aim of this section is to explore the bandwidth of

70 Alfred Mann, "The Study of Counterpoint from Johann Joseph Fux's Gradus ad Parnassum."

71 Leonard J. Paul. *Advanced Topics in Video Game Audio*. Invited Talk, Stanford University, CCRMA, May 2015. https://www.youtube.com/watch?v=tvx_QgS7D8Q, accessed, May 10, 2018.

possible interactive parameters further, characterise the musical flexibility more precisely and provide insights on the practical limitations of parameter modulations. Experiments have been undertaken with two conceptually different music engines, the script-based *AmbientMusicBox* and the less deterministic, highly parameterised *AmbientMusicGenerator*.

Ambient music, thanks to its compositional paradigms that often involve concepts of quasi-serial and minimal music, seems a natural fit for these investigations. Its relevance to game scoring has been pointed out previously. Ambient music *is* frequently used in video games today. Hence, the technical, as well as the musical framework, of these experiments should provide insights of practical relevance.

AmbientMusicBox is a real-time script interpreter and synthesiser. Its music is defined via a dedicated XML-based scripting language. The main aim for the definition of the application programming interface (API) was to provide simple, high-level music controls to the game engine and game developer, respectively. Hence, on the application side no in-depth musical knowledge should be necessary to integrate *AmbientMusicBox* into the game mechanics. The set of commands comprises the loading of music scripts into working memory, triggering playback start and stop, channel gain settings, and session recording. Additionally, a synthesiser for wind sounds (with the parameters howl speed and strength) is offered. Audio data can be incorporated which is particularly useful for ambient noises, synthetic as well as naturalistic soundscapes and sound effects.

The scripting language provides dedicated routines for music definition. The following descriptions will focus on its core elements. A more comprehensive technical overview of *AmbientMusicBox* and its scripting language is given by Berndt et al.[72] The structure of the music script is as follows. The root node `<music/>` holds one or more `<voice/>` elements. These represent individual instruments. The structure of a voice element is:

72 Axel Berndt, Simon Waloschek, Aristotelis Hadjakos, and Alexander Leemhuis, "AmbiDice: An Ambient Music Interface for Tabletop Role-Playing Games," *Proceedings of New Interfaces for Musical Expression (NIME) 2017*, (Aalborg University Copenhagen, Denmark, May 2017).

 10.5920/beyondairports.fulltext

```
<voice          channelLeft="" channelRight=""
                name="" instrument="" polyphony=""
                fadeSpeed="" relativeMix=""/>
```

Each instrument and DSP effect (specified as child of `voice`) can have a mono or stereo output (specified via `channelLeft` and `channelRight`). The attribute `name` specifies an identifier string that can be used by the application to address the voice, e.g. for dynamic mixing. The attribute `instrument` sets the sound synthesis algorithm to be used for this voice. *AmbientMusicBox* offers a set of prefabricated patches that can be further extended by custom synthesis and effects patches. Attribute `polyphony` specifies the instrument's polyphony, the number of copies to be instantiated to play several notes at the same time. With `fadeSpeed` volume changes are ramped, `relativeMix` sets an initial volume gain (between 0.0 and 1.0). In case of an audio track, audio data is loaded inside the voice. Otherwise, the voice is an actual musical instrument, i.e. it plays notes. Each such instrumental voice specifies one or more sequences to be played, such as the following example.

```
<sequence       loop="inf"
                maxStartDelay.seconds="10">
        <note   pitch.midi="70" velocity="0.4"/>
        <rest   dur.seconds="4"/
        <note   pitch.midi="75" velocity="0.3"/>
        <note   pitch.midi="71" velocity="0.5"
                dur.seconds="25"
                dur.variation="0.025"/>
        <rest   dur.seconds="4"
                dur.variation="0.025"/>
</sequence>
```

In the case that more than one sequence is defined, they are performed simultaneously. The attribute `loop` is set to a nonnegative integer value or `inf` to specify how often the sequence is repeated. The whole concept behind sequences accounts for the typical approach of composing ambient music from (asynchronous) event cycles. With the attribute `maxStartDelay.seconds` the beginning of this sequence can be delayed by a random floating point value (in seconds) up to the given value (10 seconds in this example). The basic building blocks of sequences are `note` and `rest` elements. Their durations are specified in seconds. This seems counter-intuitive in the musical context, but a peculiarity of many ambient compositions is the absence of musical meter in a traditional sense. For this, it is easier to work with absolute timing. Durations can also be subject to random variation which is achieved by the attribute `dur.variation` that defines a maximum variation radius (in both directions, positive and negative). Sequences may contain further sequences that can be played in succession or simultaneously. The fourth possible type of child elements in a sequence is `procedure` in one of the following three forms:

1. `<procedure mode="random choice"/>`

One of its child elements is randomly chosen and performed.

2. `<procedure mode="permutation" numberOfPermutations=""/>`

When the procedure is processed the first time, the first child element is performed. At the second iteration the second child element is performed, and so on. After all children were performed, the specified `numberOfPermutations` is applied to the series and the playback starts anew with the new first child element.

 10.5920/beyondairports.fulltext

3. `<procedure **mode**="permutation sequence" **numberOfPermutations**=""/>`

This type of procedure has only one child element of type `sequence`. The first time the procedure is processed the whole sequence is played as is. From then on, the given `numberOfPermutations` is applied to all non-rest children of the sequence before it is performed again.

Procedures are the major means for introducing nondeterminism to the music. They may contain notes, rests, sequences and further procedures. Nesting of sequences and procedures is a powerful tool to build up complex compositions from these rather few musical building blocks.

From the application's perspective *AmbientMusicBox* provides deterministic and nondeterministic music that is capable of playing for a long time without relying on repetition. Channel gain controls offer an effective interface for dynamic mixing. Technically, it is also possible to access the music scripts at runtime and alter the music on the compositional level. This requires the application programmer to have the necessary musical background or a more intimate involvement with the composer during the development process.[73] In this way, reharmonizations can be implemented. However, more drastic alterations to the scripts are rather expensive from both the compositional and algorithmic perspective, hence, impractical. *AmbientMusicBox*'s reliance on predefined compositions (the XML scripts) - even though nonlinear in themselves - and minimalistic API proved detrimental to its musical flexibility. API extensions and the addition of algorithmic tools for more complex transformations and seamless transitioning between scripts are essential to obtain further flexibility.

AmbientMusicGenerator is a real-time algorithmic composition engine. Instead of relying on predefined compositions (such as *AmbientMusicBox*'s scripts) it is interfaced via a series of musical parameters. These are scanned

73 In contrast to this, music production is often subject to outsourcing and takes place at a late stage of game development.

whenever new musical material is generated. Hence every parameter change will immediately result in a change of one or more musical features. The following discussion will focus on these parameters and their modulation. A complete description of *AmbientMusicGenerator*, its algorithmic approach to ambient music and technical implementation is given by Maxim Müller.[74]

First of all, sound synthesis is done in real-time again. It is possible to switch between predefined sounds and modulate basic synthesis parameters such as attack, decay, sustain and release values of the amplitude envelope. This allows for adjusting each tone's articulation. The tuning/transposition of an instrument, a reverb effect and lowpass filter have an effect on its timbre. Each voice also has a volume control for dynamic mixing.

Four polyphonic musical voices are available, each with its own role and corresponding behaviour: an arpeggiator and a chord voice, a bass voice and a melodic phrase generator. They all comply with a given tonality; a defined scale, tonic pitch and main chord. Twenty scales are available, including major, several variants of minor, chromatic, church modes, pentatonic, just pentatonic, and whole tone scale. The main chord comprises up to five inversions. Within this, any position, chord inversion and voicing (close, open) is possible.

Each musical voice's activity is conceived around Eric Tamm's notion of "regularly or irregularly repeating events or cycles of events" and "breathing"[75] in ambient music. This is implemented as phrasing in the following form. Periods of activity (a certain number of sound events played with a certain density and more or less variation in their metric order) are followed by periods of silence (this "sleep time" can also vary by a random amount). The relationship between a voice's play time and sleep time, event density and rhythmic variation determines its pace. Phrases can be long and regular with

74 Maxim Müller, "Interactive Ambient Music Generation," *Works in Audio and Music Technology*, ed. Axel Berndt (Dresden, TUDpress, 2015), 59-95.

75 Eric Tamm, "Brian Eno: His Music and the Vertical Color Of Sound."

 10.5920/beyondairports.fulltext

literally no rests in between. Others can be short, dense, and arhythmical. A global tempo parameter can be used to scale the phrasing of all voices at once while keeping their mutual relationship intact.

Each voice (apart from the bass that plays random notes from a given chord) has further individual parameters that relate to its musical role.

- The arpeggiator voice generates an initial random figure and applies permutations to vary it. A parameter can be set to vary the frequency at which this happens;
- The pitch range of the chord voice can be specified. Strictly synchronous note onsets can also be broken up into a looser, more arpeggiated playing technique;
- The melody voice generates its melodies from a Markov model. It can be retrained on the fly by changing the training set (the set of sample melodies) and the order of the model (the length of melodic patterns to learn from these melodies). This causes corresponding changes of melodic features in the music generated from it. Rhythmic features can be included or excluded from training. The melodies that are generated, or more precisely the sequences of pitch intervals,[76] employ only pitches from the underlying scale which can be further limited to pitches only from the main (tonic) chord.

The rhythmic part of the melody voice can be applied to all other voices, too. The other voices will then feature similar rhythmic properties.

Thanks to its highly parameterized algorithm, *AmbientMusicGenerator* offers musical flexibility on several levels: dynamic mixing, expressive performance, reharmonization, and in compositional aspects such as phrasing, melodic gesture, harmonicity, and rhythm. Some parameters may have a subtler effect when modulated, others are clearly audible. Even though all output complies with the ambient-specific conceptualisation that reflects

76 This is a prerequisite for being invariant with transposition and scale.

in the algorithmic approach, *AmbientMusicGenerator* offers more flexibility than all other approaches discussed so far.

When experimenting with on-the-fly parameter changes in *AmbientMusicGenerator*, it becomes immediately apparent that there are essential aesthetic conditions to be considered. Drastic and abrupt parameter changes result in likewise incoherent musical changes. It is advisable to transition these over a certain period of time so that the musical transformation actually becomes a process of musical development. Most numeric parameters can simply be ramped by linear or nonlinear interpolation over time, starting with the initial value and leading to the target value. In the context of sound synthesis and dynamic mixing, this is common practice. But it also applies to phrase length, event density, tempo, frequency of permutation of arpeggio patterns, and pitch ranges - basically all musical aspects that can be abstracted as bipolar numeric ranges.

Other aspects, however, require more dedicated treatment. Melodic transitions can be achieved by piecewise changes of the training set, i.e. adding a few new melodies while removing some that are not in the target training set, then retraining the Markov model and repeating the procedure every few time frames until the training set is in its target state. The main chord can be transitioned via classic harmonic modulation techniques or in a simpler way by changing only one chord position at a time. Timbres might be interpolated if they use the same synthesis patch. Otherwise it is necessary to switch to another timbre which is best achieved during sleep time in-between two phrases and, again, not all voices at once.

A practical limitation comes to light when considering the sheer quantity of the all too specific musical parameters that the API passes through to the application. It requires a musically competent application developer to handle them which is not always a given. It is also relatively intricate to control all of the parameters individually, especially when the aim is to transition from one musical setup to another and not just apply a few

 10.5920/beyondairports.fulltext

changes. Hence, it is advisable to introduce some high-level controls as well. *AmbientMusicGenerator* offers three such high-level controls.

1. It is possible to define presets, i.e. predefined parameter settings that the application can trigger with one convenient function call. These presets do not have to be created by the application developer but can be delivered by a composer who would also be responsible for the definition of sounds/timbres;

2. A scripting interface allows for the definition of parameter modulations over time[77] and stores them for reproduction. This is where composers can define musical pieces that do not just rely on static parameters but constitute overarching structural properties from parameter changes (e.g. an A-B-A form where A and B represent different parameter settings). These scripts can also be used to define transitions. The composer would specify how to transform one musical state/preset into another. The application only needs to execute these scripts instead of tweaking the parameters individually (which is nonetheless possible);

3. Keeping a musical state or preset playing for a long time can become monotonous, even if it is not repetitive. The above mentioned scripts might be used to implement some predefined variation. Another way of introducing more overarching variation is offered by stochastic parameter modulations. Starting with a certain preset the application specifies the amount and frequency of parameter changes. *AmbientMusicGenerator* will then choose a parameter at random each time and change its value within the given bounds. In this way an otherwise monotonous music starts to develop over time. The application controls how quickly this performs.

77 e.g. "At time *t* start transitioning parameter *x* to value *y* over the course of *n* seconds."

The fact that *AmbientMusicGenerator*'s parameters are directly accessible to the application opens up a further interesting perspective to couple music and interactive context. It is now possible to use any data stream to modulate one or more parameters automatically. This could be, for instance, the distance between player position and level exit, the number of non-player characters in the scene, the complexity of a puzzle, health meter, mouse cursor movement, or sensor data (e.g. light, accelerometer, microphone). Music now functions as a sonification of this input data, a Musical Auditory Display. Of course, the input data has to be preprocessed to be mapped to a musical parameter, typically by scaling the range, adapting the resolution (quantisation) and removing undesired frequency content (preventing parameter changes that are too abrupt, for example). Such a Musical Auditory Display establishes a much closer relation to player interaction and the game's diegesis. It can help to create an even more personalized player experience.

Conclusion

The omnipresence of music in video games deserves careful consideration not only with regard to its narrative role but also its implementation or, more precisely, technical integration with the games' mechanics. Nonlinear media demand likewise nonlinear music. The bandwidth of possible approaches to such an adaptive, nonlinear musical score reaches from arrangement techniques over on-the-fly alterations of its expressive performance, reharmonization, embellishment, and improvisation to generative transitions and, ultimately, real-time procedural composition.

In today's games, dynamic sequencing and mixing are well established. They can be achieved with static audio data and standard functionality that modern game audio engines typically provide out of the box. They integrate well with the established workflows of composers and game music studios. But the musical flexibility is limited to what this material offers, i.e. what has been composed and produced in advance.

 10.5920/beyondairports.fulltext

More flexibility can be attained by altering musical data in a more differentiated way, especially by generative performance, variation and transition techniques. These, however, have to stay true to the aesthetics of the pre-composed material in order to satisfy. Encouraging results have been demonstrated in Baroque and jazz style contexts. But the ideas and their current implementations are still mostly academic, prototypical and far from ready to use in commercial gaming media. Automatic adaptation of pre-composed music is a nontrivial, challenging task that needs more thorough theorisation in the coming years.

Challenging aspects of the aforementioned approaches arise from the fact that they deal with pre-composed music, typically created by human composers. Their aesthetic agenda is usually not fully formalised, if at all, which makes it hard to process their music appropriately. In the world of procedural or generative music things turn out differently. The algorithms behind this kind of music are, in fact, not only full formalisations but at the same time supply various aesthetic parameters to tune their output. Using the example of ambient aesthetics - which is highly relevant to game scoring - it has been discussed how its parameters can be modulated by game mechanics in a musically satisfying way and how its full expressive range is made accessible to game engines.

Such musical flexibility, however, does not come easily. It requires a rethinking on many levels. The composer no longer delivers temporally-fixed pieces but "musical landscapes and mechanisms". Composing and producing these has nothing in common with classical music (record) production but requires different methods and workflows. Just like interactive graphics, nonlinear music arises and develops in the moment of its performance. Hence, in the same way as real-time graphics rendering, real-time sound synthesis must become a common element of game engines, including an appropriate share of memory and processing power. At best, game developers are willing to invest development effort in a music engine capable of creating an overall coherent score and playing experience.

However, not every video game needs such musical flexibility. Game developers should ask themselves which aspects of the game they want the music to reflect and react to, how quickly they want the music to react, how much time their players are going to spend in certain scenes or game states, for how long should the corresponding music play, and ultimately, which musical approach serves their goals best without overcomplicating things.

10.5920/beyondairports.fulltext