Synthetic game audio with Puredata

Andy Farnell (andy@obiwannabe.co.uk)

August 24th, 2007

Abstract. This document looks generally at the application of synthetic audio to video games and presents some experimental results and analysis from research in generating efficient synthetic sound for virtual worlds. Dynamic sound effects for real-time use in game worlds require far more careful thought to design than brute force methods employed where sound is computed prior to delivery, not just in terms of static computational cost, but in how they are structured and behave when parametrised by world events. To make synthetic audio a possibility for the next generation of game audio compromises must be found that satisfy several conflicting variables. Examples are presented that demonstrate cost effective approximations based on key physical parameters and psychoacoustic cues, what can be called "Practical synthetic sound design".

1 Why the move towards synthetic sound?

Some may argue that synthetic audio has always been a solution looking for a problem, that synthesisers were just fashionable musical toys in the 1980s. Now a real problem has arrived. That problem is how to provide the colossal amounts of content required to populate virtual worlds for modern video games. While graphics technology has received an enormous investment of money and attention sound, despite claims in defence of the games industry [2], is still in the dark ages in terms of technology. The reasons for a reluctance to embrace progressive technologies deserves an entire study by itself, which I will address elsewhere.

1.1 Generating content

Anecdotally, it seems as of Autumn 2007 many directors and project leaders in game development are realising it is challenging to generate sufficient content to populate worlds even with a full-time team of sound designers. The size of games worlds and the number of objects will continue growing. One solution for massively multi-player worlds such as "Second Life" is to rely on user generated content. Disadvantages to this, are unreliability, inconsistency and lack of appropriate control. In any case it is desirable to have default sounds for most classes that are overridden by user content. Massively single-player models such as "Spore" rely on procedurally generated content. There are important differences between procedural content and synthetic content which are subtle, and will be discussed later, but essentially synthetic audio is an extension of the "massively single-player" approach applied to the general case of virtual world sounds.

Growth The mapping between objects and their visual manifestation is one to one. Although many instances of a particular object may be created, for each instance there is one mesh and one set of textures that defines its appearance. Regardless of the distance from which it is viewed or the lighting conditions the same data is used to represent an object. By contrast, sound is an interactive phenomenon, so there is not a fixed number of sounds belonging to one object. If you like, the number of "object views" is greater than for visual appearance.

Relationships If we take the entity relationship "collides" to be unique per object pair then the growth is factorial. Of course this is overkill, for the majority of relationships such as `hammer->hits->bell` or `anvil->hits->bell` are indistinguishable. But the number of realistic couplings that lead to distinguishable excitation methods and consequent sound patterns is still large. An object may by struck in a glancing fashion or impacted by a body that remains in contact, it can be scraped to cause frictional excitation, blown to excite an air cavity or shaken by a nearby coupled source into resonance. These things can happen in air or underwater affecting the speed of sound in the medium, or while the object, observer or propagation medium is in motion causing Doppler effects. Additionally, any sound object may be parametrised by a set of continuous physical variables far more extensive than the small number of vectors defining appearance.

Combinations Consider for example the case of footsteps in traditional game sound design. Conservatively we might have 20 surface textures such as wood, metal, gravel, sand, concrete and so forth. We must multiply this by the number of characters that have unique footwear, perhaps 3 or 4, and again by several distinct speeds of movement since a footstep sound is different when the player creeps, walks or runs. We might also like to factor in the weight of the player according to personal inventory, and maybe adapt the footsteps to reflect the work done ascending or walking on a level surface. Taken as discrete states this matrix of possible sounds quickly grows to the point where it is difficult to satisfy using sample data. Difficulties arise with consistency of level and quality as the data set grows. We get to the point where processing is better deferred until performance (play) where continuous rather than discrete control is possible.

Access A further problem arises with the length of play and arrangement of the audio data pool. Some adventure games may extend beyond 40 hours of play. "Heavenly Sword", a recent PS3 title boasts over 10GB of audio data. 10GB of data requires careful memory management, placing data at reasonable load points and forcing the flow of play to be over-structured. This moves us towards a linear narrative more like a film than a truly random-access "anything can happen" virtual world. From many points of view we are approaching a situation where it may not be possible to continue extending game audio content within the data model, even with advanced storage like Blu-Ray or HDVD. The storage requirements are dwarfed by data management requirements.

1.2 Parametrising content

Some variations of sound, traditionally handled by blending loops of given states, are hard to recreate with the data model. Composite objects such as an vehicle engine or other machines may have many sounds that are ascribed to the same source depending on continuously variable parameters rather than events or states. The sounds of an engine starting, accelerating, stalling or misfiring are all different, yet come from the same source. The sound of an engine close to the cylinder block, inside the vehicle, at the rear close to the exhaust, or the sound of a distant vehicle speeding away are all different manifestations of the same object. To maintain a continuous relationship between all these positions and states while keeping the components synchronous for blending is difficult. Techniques developed for manipulating sample loops [10] are very poor in comparison to synthetic methods and since this has been a recurring complaint from players and developers alike we are keen to encourage developers to look at embedding code for vehicle engine synthesis.

Dynamic and varied content "Dynamic" in this case does not refer to the range of loudness but to the ability to reconfigure the DSP during performance, for example to dynamically alter a filter as an object moves around. Sample data intrinsically lacks dynamics, it is a fixed recording of a sound. Research has been conducted and articles written [15] on avoiding too much similarity when using sample data. Many of these amount to introducing random playback points, replay speeds and filter settings. These can really only be seen as temporary remedies or tricks. A major strength of synthetic sound is introducing variations that mean no two sounds from the same event need ever be exactly the same. This is an inbuilt feature of subtractive methods that use noise as the basis waveforms. No two examples of the same impact using a noisy excitation and material formant approach will ever have exactly the same time domain waveform. Although the differences may be extremely subtle, or even immeasurable by spectral analysis, the brain has a peculiar ability to recognise that these sources have some "live" quality compared to exactly the same waveform replayed over and over using sample data. Although I have no hard experimental evidence to support the claim it seems they are less fatiguing. Sometimes it is desirable to introduce subtle variation into sound events that are very similar but distinguished by a continuous variable. An example is a dropped ball which sounds very slightly different on each bounce as it loses kinetic energy to entropy. Some objects like bells and tubes create different sounds depending on their state. A tube that is already vibrating and is then struck in a different location incorporates the new excitation into its vibration pattern to make a different sound than if it were struck while at rest. This is mentioned later where we consider the advantages of waveguide models, though I would like to note here that it is not possible with sample data because superposition cannot reproduce it.

2 Synthesis Methods

Traditional approaches to synthesis tend to focus on one method and attempt to use it universally. Here I am concerned with efficient results rather than an attempt to explore the theoretic limits of any particular method. Theoretically the range of timbres obtainable from each method is equivalent, but in reality there are boundary conditions that limit each to a practical range. For example, additive and subtractive methods are complementary, each containing a special case that is extremely expensive in one method while trivial in the other. The practical approach is to use DSP built from hybrid methods, to use all the tools in the box. That means not being limited to spectral, physical, or time domain waveform modelling, but rather to use each as it is appropriate. Physical and psychoacoustic interpretations are also very important. They represent the end points in a synthetic facsimile, the cause or production mechanism is rooted in physics, while the effect is psychological and perceptual. Many solutions in practical synthesis involve some kind of "trick" or efficiency. Some of these are undocumented, but the majority are taken and used creatively from literature on psychoacoustics [12].

2.1 Synthesis Implementation

Our synthesis methods are developed in Puredata. While dataflow systems cannot express all idioms required to make any kind of DSP, for example it lacks recursion and effective sample accuracy to implement IIR filters, it is extremely productive, easy to to work with and covers 90 percent of requirements for synthetic sound design. Because games developers have never adopted such a thing as a real "sound engine" (meaning a system capable of executing arbitrary DSP graphs), it is currently necessary to rewrite code using Faust [17], STK [4] or hand craft the prototypes in C/C++. Hopefully tool-chain improvements will soon make it possible to write synthetic sound objects directly in dataflow for execution on game consoles. Nevertheless, current design approaches must look ahead as if such a path were already available, so many programming techniques are used in anticipation of real-time client-side execution. At all stages efficiency is paramount. One constraint is to avoid the use of memory operations as much as possible. Wherever possible an efficient computational solution is preferred to one involving a lookup table. Most algorithms start with one or more phasor blocks, an accumulator from which other periodic functions can be derived. Sinusoidal and cosinusoidal functions are efficiently calculated using a truncated Taylor series approximation, or a reflected 4 term parabolic approximation. Noise is obtained by linear congruential pseudo-random generation. Waveshaping operations such as Chebyshev tables can in many cases be substituted by polynomial expansions limited in range to the areas we are interested in. The reason for avoiding memory table structures is to facilitate future execution with parallel threads across multiple processors where it is easier to manage numerical code expressed in a purely procedural way without reference to shared resources. Simple FM and AM methods that do not require "musical accuracy" are quite effective for sound effect design. As given by Arfib and LeBrun there is an equivalence between FM and other non-linear methods. Hybrid non-linear synthesis methods that can be interpreted partly as waveshaping and partly as modulation are attractive because they can yield natural spectral evolution with great efficiency. The disadvantage of these is that they must often be designed specifically for a narrow class of sound objects, such as pipes, metal sheets, strings and so forth, and are not easily programmable as general purpose sources.

2.2 Physical Modelling

The disadvantage of waveguide methods [5] is their requirement for memory in the form of circular buffers for propagation delays and resonators. Mass-spring-damper networks are efficient in terms of element cost but require an extremely large number of elements to approximate realistic sounds. Again, a practical compromise is to combine waveguide or MSD methods with non-linear ones to construct hybrid structures which take the best of both approaches while offering efficiency. A waveguide tapped at critical points or MSD oscillator comprising a small number of elements may be substituted as the primary oscillator in a hybrid system to add "statefullness ", by which the sound of an object depends on its previous level of excitement. This works well for struck tubes and bells and "dropped" sounds where objects undergo a large number of collisions at different points on the body within a short time interval during bouncing or rolling.

2.3 Granular approximation

Granular synthesis is appropriate for a class of sounds that are heterogeneous extents. Examples are water, waves on the seashore, rain and crowds of people. In the case of crowds, one example investigated is applause. Individual clapping is easily synthesised by carefully filtered noise bursts. Difficulties arise when combining a large number of these. Statistical analysis of density [14] can help with getting correct sounding results but it is hard in practice to achieve a consistent and realistic composition from summing individual claps efficiently. It was found that beyond a certain density, about 20 claps per second, blending in granular noise with the correct spectrum in order to blur the composite effect gave great results at low cost.

2.4 Dynamic DSP graph construction

An interesting advantage of the dataflow system given by Pd is that instead of sending entire synthesisers as sound objects to the client we can build new ones at a low background cost by sending instructions to insert and connect new atomic objects. This is a big advantage over the previously proposed MPEG4.SA mechanism for delivering synthetic sound to a remote client. Instead of sending unit generators or entire synthesisers we send procedural "code to build code". Unfortunately it isn't yet possible to do this on a 'live' process. Ideally we would like to be able to construct DSP chains dynamically, adding in unit generators or new abstractions and crossfading these into the running sound once stable. Unfortunately a current limitation of Pd, which is built on a doubly linked list datastructure, is its requirement to rebuild the entire DSP graph when new units are inserted which causes clicks and audio dropouts. Some thought is presently being given to this problem by Pd developers.

2.5 Voice management

The ability to abstract DSP operations and instantiate parallel or cascade forms in a single operation is powerful. Supercollider and Fausts DSP algebra are both able to do this but neither replaces Puredata for efficient real-time execution, the latter being an intermediate stage for producing compilable C++ code. While it currently has some shortcomings Barknechts [nqpoly4] unit is a powerful device for achieving simple parallel voicing for granular synthesis. Techniques borrowed from music technology are quite appropriate for most sound effects, which are either sustained or one-shot decaying patterns. DSP code may be turned off and deallocated once a sound has decayed beyond a threshold value. Polyphony of can be handled by voice stealing from least active or quietest instances, or simply allocated round-robin from a fixed pool.

2.6 Dynamic level of detail

We wish for sounds to grow or diminish in detail with proximity, atmospheric damping and occlusion as the player or objects move around. In traditional game audio technology such as the FMOD engine, simple attenuation or low pass filtering is applied as a function of distance. Synthetic sound wins out over data driven sound as we approach the condition of large scale sound scenes, between 100 and 1000 concurrent sources. Sampled data retains a fixed cost per source, but the density of a typical scene has a limit, both in the practicalities of accurate summation and the psychoacoutics of superposition. At around 100 signal sources the effect of contributory sampled signals for their cost diminishes. On the other hand a synthetic solution can offer variable cost per source. An example of this is breaking glass where we model each fragment as it falls onto the ground. A synthetic method can produce very realistic results and still retain a high degree of audio visual correlation by replacing the peripheral fragments with single sine or noise granules while providing a high level of detail for fragments that fall close to the player.

3 Examples

I have scores of examples of efficient real-time executable sound effects at various stages of development. Some of the more advanced synthetic sound objects in development include animals such as birds (from [16], [13], [11], [7]) and mammals, weather features such as rain and thunder (from [9]), or particular machines such as cars or aeroplanes. Most of these have specific features which are too detailed to discuss here and I am constantly exploring innovative new ways to parametrise them according to more detailed physical interpretations. The five examples below are given because they illustrate one or more general features that are common to other models or general principles that have emerged during research.

3.1 Material and structure

Once class of sounds accounting for the vast majority of incidental audio in any scene is excitations of fixed structures. Dropped objects, bullet impacts, doors closing and so forth. Noisy impulses applied to networks of parallel and cascade filters may be used to approximate the static formants of almost all common materials. In thinking about how to synthesise materials generally it is necessary to consider the propagation of energy within it. Ideally large MSD networks would be used, but these are too expensive so we need to employ less expensive filters with data from spectral analysis and some heuristics. There is some correlation between bulk modulus, density and damping. The exact damping of a material is not easy to predict from it's chemical composition, it also requires knowledge of the microstructural composition. For example, wood and cardboard are both cellulose fibres but have different densities and arrangements of material. Textbooks of physical and material constants are helpful in picking numbers, but occasionally it is necessary to just experiment with real materials and a spectrograph. Sound file [1] gives metal plate, card, wood, glass and metal formant modelling examples.

Propagation and damping Energy is reflected more in elastic materials such as glass than in wood, with metals lying somewhere between. Data has been taken from uniform excitation of plastic, cardboard, wood, china, metal and glass with equal geometry in order to create cheap filters that approximate these materials. Of course it is not possible to separate geometry, and eigenmodes of propagation, from the absolute frequencies found in any sample. What matters is the ratios of each frequency and the ratios of decay which are

[1]http://www.obiwannabe.co.uk/sounds/effect-pmimpacts.mp3

set by filter resonance. To a certain degree these simple formant filters can then be scaled to account for larger bodies of the material. Formant filters of this kind may be used as coupling materials to construct composite objects. An common example from musical instrument literature is to insert a virtual bridge between the string and sounding board of a violin, but this technique can be used generally to create many sound effects by considering the physical construction of an object.

Telephone The example given is a telephone bell[2]. What makes this sound effect work more than anything else is the crude approximation to ”bakelite”, a thermosetting resin used to construct old style telephones. The bells alone, which are additive models, sound reasonably good, but the sound comes alive when we add a short noise burst to approximate the solenoid armature coupled to the body of the telephone. Composing sound by thinking about the physical coupling of materials leads to coherence. The second example is a creaky door[3] which comprises two parts. The first part is a squeaky hinge and the second part is the slam of the door in its frame. In reality the hinge is attached to the door, so it makes a sound that is strongly affected by the properties of the door not just the hinge. By making the squeaky hinge resonate through the same filter as used for the door we bind these sounds into something that is recognised as the same object.

3.2 Fire

Fire is great example used to demonstrate dynamic level of detail[4]. The full model itself comprises a component analysis of the physical processes during combustion of wood. Different models are used for gas and liquid fires, but the ”bonfire” example covers all of the properties we are interested in. It is an example of a heterogeneous or composite sound, comprising many different sources that together create the desired effect. These, (with the common English word used for the sound in parentheses) are, fragmentation of fuel (crackling), outgassing of water vapour and other gaseous phase compounds (hissing), aerial conflagration of particles (fizzing), turbulence at the gas combustion front (roaring), settling of fuel (clattering), quasi-periodic relaxation oscillation during outgassing (whining), and low frequency air induction (woofing). In addition we may consider some more processes such as stress caused by thermal expansion of fuel and nearby objects (groaning) and boiling of liquid phase components (sizzling, bubbling). Each of these components may be approximated with its own unit generator, the details of which are too complex to consider here, but the essential feature is that they are causally linked and blend in a particular way according to the intensity of the fire and the distance of the observer from it. The example shows a simplified model of fire that would be used at a medium level of detail. It contains the hissing, roaring and crackling elements.

[2]http://www.obiwannabe.co.uk/sounds/effect-oldphonebell.mp3

[3]http://www.obiwannabe.co.uk/sounds/effect-doorcreak1.mp3

[4]http://www.obiwannabe.co.uk/sounds/effect-fire.mp3

3.3 Wind

Wind is an example of implicit production. Wind makes no sound at all, it is other objects in the path of moving air flow that make the sound of wind. Therefore every object in a scene has the potential to make a sound caused by turbulent flow around it. This would be impractical in a real scenario and raises the interesting question of how to limit objects that have universal implicit production methods. The boolean qualifier ”Can” might be applied to objects that `Can_Burn`, or `Can_Fragment`. They represents a threshold qualifier that is evaluated before investigating any degree of combustibility, fragmentation or ability to howl in the wind. Objects that `Can_Howl` might be telephone poles and wires, rough surfaces of brickwork or large rocks. So wind is a distributed extent that emits from many objects in the location of the player. The key to a good wind sound is to model this parallelism, often with just 4 or 5 sources within a certain radius of the player. Wind sounds themselves [5] are extremely simple, band-passed noise with a fair degree of resonance and a center frequency that maps to the air velocity. Narrow objects like wires tend to emit higher and more resonant sounds, they whistle. Large irregular objects like rocks form the unfocused rushing sound of the wind. The psychoacoustic trick to really nice wind is to recognise that although each emitter moves in parallel there is a non-trivial propagation time as the wind moves across the scene. If the wind direction is left to right then a slight delay in the rise and fall of sources on the right side relative to the ones on the left produces the correct effect. This is achieved by simply low pass filtering the control signal.

3.4 Water

Flowing water is a homogeneous extent, being composed of many asynchronous sources that approximate to a rising sine wave distributed across a surface. These

[5]http://www.obiwannabe.co.uk/sounds/effect-wind2.mp3

are the result of air resonating in small cavities produced as the surface is disturbed. The body of water itself imparts some resonance to the sound and indicates depth. The density of the sine wave components is a good indicator of speed of flow. We can approximate most bodies of flowing water with three variables, depth, speed of flow and impedance. The last of these is a measure of how much obstruction the liquid encounters causing irregular movement and cavities. There are several interesting ways of synthesising water such as subverting noise reduction algorithms, taking the FFT of noise and thresholding out all but a few of the mid range components before resynthesising with an inverse FFT. A less expensive way is to employ an additive/granular method which composits many small bursts containing a rising sine wave. The example given [6] is an efficient algorithm developed by trial and error that takes the positive part of the first difference of low pass filtered noise and uses it as a frequency modulator for a sine oscillator.

3.5 Footsteps

This is an example which demonstrates the range of real-time parametrisation possible [7]. It is based on an analysis of bipedal movement. Actor speed and weight are combined with a footprint pattern to obtain a ground response force (GRF) which approximates the pressure at each stage of a step. There are three parts for each step, one for the ball of the foot, one for the heel, and one for the outstep. On each step the weight is transferred from the heel to the ball in a pattern that depends on whether the actor is accelerating or slowing down, moving uphill or on the level, because different amounts of work are done in different parts of the foot [3], [6]. As the player moves between different speeds the phase relationship of six parts, three on each foot, changes. Running involves a quite different sequence than creeping. The former maximises locomotion, the latter, which is a feature of predators, minimises the change in GRF for stealth. Walking is a compromise that trades off efficient locomotion against energy expenditure [8], [1]. These control cycles have been modeled using polynomial approximations of the GRF curves and fed to a simple granular synthesiser that gives reasonable dirt and gravel surface textures. With a set of synthesisers appropriate for different surface textures this arrangement can replace many megabytes of sample data with a few kilobytes of code while giving subjectively superior results.

[6]http://www.obiwannabe.co.uk/sounds/effect-water1.mp3

[7]http://www.obiwannabe.co.uk/sounds/effect-footsteps-gravel.mp3

3.6 Machines

Some machine sounds are ambient keypoints, from the players point of view they are always running and always have been running. Noises such as fans, ventilation blowers, refrigerators and generators are part of the furniture in a world level to add a background ambiance. In the simplest interactive case a machine, weapon or tool always has at least one event hook to "activate". This is usually a boolean toggle and implies deactivate. This immediately adds three other considerations, a switch sound to activate/deactivate, and start and stop versions of the continuous running sound. It implies a transition between the two states, one for starting up and one for stopping. Machines that have higher degrees of control may require several continuous variables, and of course these are unique to the object in question and can't be discussed here generally. At least one will usually be some kind of speed/intensity control. When speed or intensity of a variable change there is often a secondary effect, for example a machine acts differently under changing load while the motor spins up than it does at a constant work rate, so we often derive $|dV/dt|$ as an automatic continuous variable. The more interesting class of machines are mechanical, because we can consider real physics. Electronic machines are entirely from the realm of fantasy and can have arbitrary sounds. It's worth noting that all machine sounds are a result of inefficiency and friction. Perfect machines would make no noise at all in theory.

Propeller Fans and propellers are modelled by a variable slope pulse wave modulating a resonant filter and short delay [8]. Sound emits from the blades as turbulence noise and is modulated in frequency by Doppler effect from the fast moving edges. Where the listener is perpendicular to the plane of rotation a more even sound is heard, but as the blade edge reaches a rotational velocity of 340m/s a chopping/snapping sound is heard where the listener is in the plane of rotation. The apparent position of the listener can be changed by altering the attack slope of the pulse wave. This is good for helicopter sounds or aircraft that move past the listener.

Electric motors Motors are modelled by considering their design, generally the brush and commutator type which give a high pitched whining sound [9]. Some sparking, which is modelled by chirp impulses can be

[8]http://www.obiwannabe.co.uk/sounds/effect-propellers2.mp3

[9]http://www.obiwannabe.co.uk/sounds/effect-robotmotors.mp3

added for good effect. One of the strongest characters that affects a motor sound is the resonance of the housing and how it is mounted to a support. If a motor is in a metal housing we use formant filters of short waveguides to simulate the body resonance. A particular subversion of FM is very helpful here, by modulating the carrier and keeping the modulator fixed in a simple FM block we can simulate the effect where the body resonates at certain speeds. This is very effective for starting and stopping sounds.

Switches Levers, switches, solenoids and relays are all impact or friction excitations that can be broken down into a simple series of clicks and resonances. The fourth power of a line segment is usually good enough as an exciter pulse to an allpass network or formant filter built according to the materials involved. Ratchets and cogs are simulated by repeating the excitation in a regular way. Again, the greatest influence on the sound is not that of the switch components themselves, but the surface it is mounted on. A lever mounted on a wooden board makes a clunk that is dominated by the resonance of the wooden panel. The example given demonstrates an alarm clock built from a synchronous arrangement of contributory clicks that have a common resonant body [10].

Mechanical assemblies Other components are modelled on a case by case basis. Virtual gears, belt drives, and reciprocating or pendulum mechanisms have all been created as the components of man made mechanical devices. The trick is to design each as a synchronous component so that they can be plugged together to create more complex machine sounds. It is easy to give a synchronous component its own clock source to make it asynchronous, but not the other way about. The example given is from the "Machine-Machine" [11], an early version of a general purpose motorised device. It features a fan, noise source, gearbox and switching sounds as well as an overdriven pipe resonance model to simulate petrol engine exhaust.

4 Conclusions

Efficient synthetic methods for game audio have been demonstrated. No single synthetic method solves all required cases and a mixture of approaches is required. Issues like dynamic level of detail and cost management have been examined and some approaches offered. Much of the groundwork for specific elements has already been done by researchers working on audio synthesis. It has been built upon here and will continue to be improved by the next generation of synthetic sound designers. Like the field of computer graphics the realism of synthesised audio will improve dramatically once a serious investment is made in the technology. With the appearance of multi-core processors the deployment of synthetic audio in games is no longer limited by resources. Programming problems such as making good use of parallelism, managing a highly dynamic DSP graph in an environment with variable resources, and automatically generating filter coefficients from geometry eigenvalues still exist. Larger scale software engineering issues allowing producers and designers to work on sound objects "in world" with appropriate tools should be a priority for practical use. A reasonable and measured approach to commercial deployment is to incrementally introduce the technology alongside existing methods.

4.1 Future directions

I believe this is a unique project which attempts to bring together the work of many researchers and add new insights. The aim is to provide the full range of synthetic sounds necessary for a complete video game and control strategies to manage them. This will take interactive sound far beyond the possibilities of currently used sample data. More work must be done to make effective demonstrations, starting with small games projects where the audio can be complemented with synthesis. The final goal of hearing a real-time generated fully synthetic game audio soundtrack using a proper "sound engine" is some way off. The remainder is advocacy and biding time for the opportunity to assemble a team capable of attacking this task. The next stage is to take this work in-house and evaluate the ability of sound designers to work with the concept of automatic generation, and using dataflow tools to construct sound objects with dynamic methods.

References

[1] S. H. Adamczyk, P. G. Collins and A. D. Kuo. The advantages of a rolling foot in human walking. *J. Exp. Biol. Vol 209*, pages 3953–3963, 2006.

[2] Rob Bridgett. Updating the state of critical writing in game sound. *Gamasutra, 31st August 2006*, 2006.

[3] Evie Vereecke K D'Aout L Van Elsacker D De Clercq. Functional analysis of the gibbon foot during terrestrial bipedal walking. *American Journal of PhysicalL Anthropology*, 2005.

[10] http://www.obiwannabe.co.uk/sounds/effect-alarmclock.mp3

[11] http://www.obiwannabe.co.uk/sounds/effect-machines6.mp3

[4] P. Cook and G. Scavone. The synthesis toolkit (stk. *Proceedings of the International Computer Music Conference, Beijing (1999).*, 1999.

[5] S. Van Duyne and J.O. Smith. Physical modeling with the 2-d digital waveguide mesh. *Proc. Int. Computer Music Conf. (ICMC'93), pages 40–47, Tokyo,Japan, Sept. 1993.*, 1993.

[6] P Aerts EE Vereecke K D'Aout. Speed modulation in hylobatid bipedalism: A kinematic analysis. *Journal of Human Evolution*, 2006.

[7] Seppo Fagerlund. Acoustics and physical models of bird sounds. *Laboratory of Acoustics and Signal Processing, HUT, Finland.*, 2003.

[8] Willems P.A. Cavagna G.A. and Heglund N.C. External, internal and total work in human locomotion. *J. Exp. Biol. Vol 198*, pages 379–393, 1995.

[9] Dipankar Roy Herbert S. Ribner. Acoustics of thunder, a quasilinear model for tortuous lightning. *J. Acoust. Soc. Am. 72*, page 6, 1982.

[10] Steve Kutay. Bigger than big: The game audio explosion. *gamedev.net, March 2007*, 2007.

[11] Frederico Avanzini Mark Kahrs. Computer synthesis of bird songs and calls. *Proc. COSTG-6 Conference on Digital Audio Effects (DAFX-01), Limerick, Ireland.*, 2001.

[12] S. McAdams and E. Bigand. Thinking in sound. the cognitive psychology of human audition. *Oxford Science Publications*, 1993.

[13] Hans Mikelson. Bird calls. *Csound Magazine*, Winter:2, 2000.

[14] Leevi Peltola. Analysis, parametric synthesis and control of hand clapping sounds. *Masters Thesis. Laboratory of Acoustics and Signal Processing, Helsinki University of Technology.*, 2004.

[15] Scott Selfon. Techniques for fighting repetition in game audio. *GDC Archives, March 2005*, 2005.

[16] T. Smyth and J.O.Smith III. The sounds of the avian syrinx – are the really flute-like? *Proceedings of DAFX 2002, International Conference on Digital Audio Effects, Hamburg, Germany, September 2002*, 2002.

[17] Stephane Letz Yann Orlarey, Dominique Fober. Faust: Functional programming for signal processing.