Even More Errant Sound Synthesis

Nick Collins
University of Sussex
`N.Collins@sussex.ac.uk`

ABSTRACT

Sound synthesis algorithms which radically depart from acoustical equations, and seek out numerical quirks at audio rate, can still have a part to play in the art-science investigations of computer music. This paper describes a host of ideas in alternative sound synthesis, from dilation equations and nonlinear dynamical equations, through probabilistic sieves, to oscillators based on geometrical formulae. We close with some new ideas in concatenative sound synthesis, using sparse approximation as the analysis method for matching, and driving synthesis through an EEG interface.

1. INTRODUCTION

This paper sets out some new recipes for quirky and abstract sound synthesis. Its remit follows the quest for new sound resources presented in an earlier work [1]. Although spectral and physical modeling have proven to have the most acoustically and perceptually realistic scope [2, 3], not all other computer music avenues have been excluded wholesale. Whilst we should take Max Mathews warnings about the limited regions of useful sound synthesis algorithm space seriously [4], neither should we cut ourselves off from potentially inspiring zones of creation. There is continued interest in the noise music community, for example, in new and wild timbral experiences, and alternative parametric means of controlling synthesis, that such alternative resources can feed directly into musical applications. Interest in nonlinear synthesis continues to bubble up online; recently with one-line C programs for algorithmic synthesis based on the simple modulo phasor/sawtooth and bit operations [5].

All of the techniques described in this paper have been implemented, with example source code and sound examples available at `http://www.sussex.ac.uk/Users/nc81/evenmoreerrant.html`, and some are already available in public open source code (such as the SLUGens in the sc3-plugins pack for SuperCollider 3); they are real, living, options for compositional application. They are also flexible to both signal processing and algorithmic composition applications, by working at different rates of generation. Sample generators at audio rate can–appropriately scaled– work well at control rates, and even as sequence generators at rhythmic rates [6].

As a compilation of ideas, the reader may read the sections of this paper in any order. Whilst some of the directions may prove profitable from scientific and technological angles, we keep an eye on arts applications, and relish the occasional nonlinearity. Whilst the bestiary-like presentation may not work in a formal journal setting, a conference paper should present some leeway for adventures such as these.

2. DILATION EQUATION SYNTHESIS

The dilation equations used in the construction of wavelets [7] can be directly utilised to create novel, self-similar, waveforms in the time domain. Recursive application of the formula:

$$\phi(x) = \sum_k c_k \phi(2x - k) \qquad (1)$$

which is equation (1) from [7, p. 615], constructs signals of increasing detail; The choice of starting signal, the number of iterations, and the coefficient array c_k lead to different scaling functions ϕ. The discrete algorithm for construction, assuming an array of N samples over I iterations, is presented in the Algorithm 1 listing.

Algorithm 1 Pseudocode for dilation equation applied on a discrete array

```
Input: $array$ of size $N$ (with initial samples), $K$ coefficients $c_k$, $I$ iterations
Output: output updated $array$
1: for each iteration from 1 up to $I$ do
2:     $newarray \leftarrow array$
3:     for each entry $val$ in $array$ at index $j$ do
4:         $sum \leftarrow 0$
5:         for each coefficient $c_k$ do
6:             $posnow \leftarrow 2 * j - \frac{Nk}{K-1}$
7:             $indexnow \leftarrow roundInteger(posnow)$
8:             if $indexnow >= 0$ AND $indexnow < N$ then
9:                 $valnow \leftarrow array[indexnow] * c_k$
10:                $sum \leftarrow sum + valnow$
11:            end if
12:        end for
13:        $newarray[j] \leftarrow sum$
14:    end for
15:    $array \leftarrow \frac{newarray}{\max(newarray)}$
16: end for
```

For safety (to avoid blow-ups in the amplitude of the y values over multiple iterations), the function is normalized to a maximum of 1 with each iteration, or at least normalized at the close of the process (assuming no numerical problems during calculation).

Figure 1 presents a collection of example signals generated by the dilation algorithm, with their parameter settings. The three central signals demonstrate the effect of increasing the number of iterations; the bottom signal is a Daubechies D4 wavelet created through this method. The original array contents, the coefficients array, and the number of iterations of the algorithm, can all be modulated over time to sonic effect. A live audiovisual demonstration has been created in SuperCollider where coefficient arrays and source shapes are dynamically interpolated; the resultant signals are depicted graphically, and sonified directly as wavetables or used to dynamically control an FFT filter's spectral magnitudes.

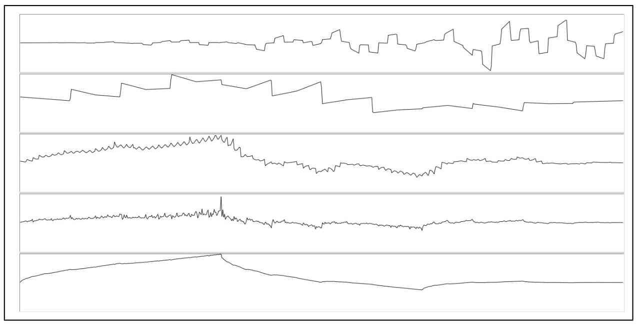

Figure 1. Dilation algorithm solutions for (1) (top line) initial array: x=y line, coefficients [1.5, -1, 2.5, 3.5, -4.5] 4 iterations (2) initial array: hat, coefficients [0.5,1,0.2, -0.3], 2 iterations (3) as (2) but 5 iterations (4) as (2) but 15 iterations (5) initial array: box (all 1s), Daubechies 4 coefficients [7, p.616], 20 iterations

3. EVEN MORE DYNAMICAL EQUATIONS

There are plenty of nonlinear equations with oscillatory solutions which have not been mined by sound synthesists. An entertainingly titled seminar paper [8] presents a selection of intriguing oscillators against a backdrop of a survey of historical European hair style changes. Whilst a number of these have been implemented in earlier computer music work (for example, Rayleigh's cubic nonlinearity for the clarinet reed has been used in physical modeling [9], the Fitzhugh-Nagumo equations were deployed in a previous publication [1]), three had not been investigated previously to this author's knowledge: the Oregonator, Brusselator and Spruce-budworm systems from chemistry and biology.

All three are implemented in the SLUGens pack for SuperCollider. However, finding effective parameter ranges can be challenging; nonlinear oscillators can blow up with certain combinations, and should be treated with caution in auditioning live. Ilya Prigogine's Brusselator equations have proved the richest sound resource in investigation so far:

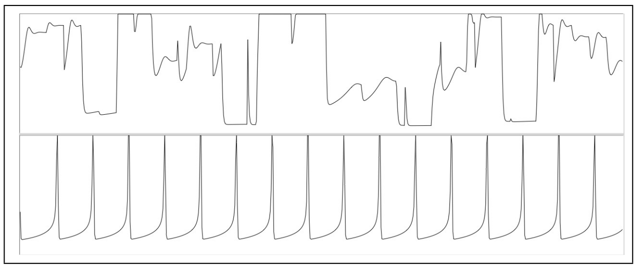

Figure 2. Two runs over 0.1 seconds of the Brusselator oscillator, with settings at top: $\mu = 0.9 \quad \gamma = 0.1$ (fall to fixed point with retriggerings from different randomly chosen starting x and y) and bottom: $\mu = 1.2 \quad \gamma = 0.2$ (limit cycle)

$$\frac{\delta x}{\delta t} = x^2 y - (\mu + 1)x + \gamma \tag{2}$$

$$\frac{\delta y}{\delta t} = -x^2 y + \mu x \tag{3}$$

An Euler ODE solver is satisfactory for low step size calculation. The Brusselator has a limit cycle region of oscillation where $\mu > \gamma^2 + 1$, or otherwise falls to the fixed point $(\gamma, \frac{\mu}{\gamma})$. In implementation, the equations' constants can be freely varied, and in practical synthesis, it is convenient to have a triggerable reset which pushes the system back to an initial state (the (x, y) starting co-ordinates of this state are available as further inputs of the UGen). Both x and y are output by the UGen, though since the fixed point is not necessarily at $x = 0, y = 0$ and outputs tend to the upper positive quadrant of the plane, synthesis can have a DC offset unless corrected with a constant addition or LeakDC filter. The two modes (limit cycle, or fall to fixed point with resets to establish pitch) give two timbral resources.

Figure 2 demonstrates time domain plots of output for x for both fixed point and limit cycle behavior. Spectral timbre is respectively a little like that of a decaying impulse train, or a rich oscillator like a sawtooth in the latter case (the waveform is like an integrate and fire oscillator); the equation parameters provide alternative control possibilities for interaction.

4. PROBABILISTIC SIEVES

Xenakis conceptualized, but never implemented, sieve based sound synthesis, as Peter Hoffman described in a recent symposium keynote [10]. He outlined a challenge to the audience to follow such methods, and I obliged him with SuperCollider code within a day, though it later transpired that Chris Ariza [11] had investigated the deterministic synthesis of sieves within a Python/Csound environment two years previously. Nonetheless, there are many interesting further variations possible from these seeds. In particular, it is productive to combine Xenakis' stochastic music, with his deterministic sieve patterns, for probabilistic sieves. Although Sever Tipei used weighted sieves more than three decades ago [12], the variant here seems original.

A probabilistic (or fuzzy) sieve is constructed as a Xenakis sieve where each element has a value from 0.0 to 1.0, rather than a hard 0 or 1 (this real number could be interpreted as a probability, or in a fuzzy conception, would be the degree of membership of the set). In practice, the array size tends to be finite rather than the implicit infinite set sizes of original sieve constructions. The infinite sieve set $\{...,-2,1,4,7,11,...\}$ or $\{3x+1 : x \in \mathbb{Z}\}$ could be written as $\{0,1,0,0,1,0,0,1,0,0\}$ if its range extended from integers 0 to 9; a related probabilistic sieve might be $\{0,0.9,0,0.1,1,0,0.1,0.87,0.01\}$. Two probabilistic sieves a and b, adjusted to have the same length (extend the shorter with 0s to the length of the longer) can be combined using various element-wise operations, [1] including:

- intersection = pq or $\min(p,q)$
- union = $p+q-pq$ or $\max(p,q)$
- symmetric difference = $p+q-2pq$

To use probabilistic sieves in sound synthesis, two options are:

- Use the probabilistic sieve as a (looping) impulse pattern generator, where impulses are triggered with probability p_i for the i^{th} entry.
- Use the probabilistic sieve to generate bit patterns (bit i on at probability p_i) for amplitude values of successive breakpoints (and possibly also inter-breakpoint interval sizes from another such sieve).

The final sieve controlling synthesis can be a result of combination operations on other simpler sieves which are modulated over time, or just directly changed itself. The Sieve1 UGen in the SLUGens uses the first approach to synthesis above, with an option for alternating sign of impulses; the latter was exhibited in language side SC code as a response to Peter Hoffman's challenge (see [10]). Probabilistic sieve patterns can also be fruitful for algorithmic composition aside from higher rate synthesis processes.

5. FROM ELLIPSE TO SUPER-TOROID

Parametric forms for geometric curves and surfaces can provide some interesting waveform shapes. Some bear relation to existing well known waveforms, but often provide alternative control parameters; the use of trignometric functions and multiplications means that formulae can often be broken down into combinations of modulation synthesis, though typically with additional operators such as exponentiation.

The parametric equation for an ellipse's x and y co-ordinates can be written:

$$a\cos\left(\arctan\left(a/b\tan\theta\right)\right) \tag{4}$$

$$b\sin\left(\arctan\left(a/b\tan\theta\right)\right) \tag{5}$$

where θ is the polar angle from the ellipse centre, and a and b are the semimajor and semiminor axes [13]. This allows us to directly synthesize with elliptical rather than circular phase progression through wavetables. In the case of direct synthesis, example SuperCollider code using this technique to synthesize the x signal (dropping the a premultiply on the final amplitude) is:

```
{
  var freq = MouseY.kr(4,444);
  var phasor = Phasor.ar(0,freq*2pi*SampleDur.ir,0.0,2pi);
  var signer = if(phasor%(1.5pi)>0.5pi,-1,1);

  cos(atan(MouseX.kr(0.01,1.0)*tan(phasor)))*signer
}.scope
```

The signer component is required to follow the quadrants of the cosine through, since the atan function is restricted to angles between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$. The Mouse control at the position of $\frac{a}{b}$, has on the right of the screen value 1, representing a circle (sine tone), and 0.01 on the left with a factor of 100 difference between minor and major axis lengths (leading to richer higher harmonics).

Although similar transformations between a smooth well rounded sine and a flattened square could be obtained using a hyperbolic tangent shaping function, the input has to be amplitude multiplied; the ellipse construction here is stable in peak to peak amplitude throughout (by having dropped the final a multiplication of the output). [2]

Beyond the basic ellipse, the co-ordinates of a superellipse are formed from powers of a sine, allowing intermediates from square wave (exponent 0.0) through sine (1.0) to squashed curves (>1.0). The Cassini oval involves a constant product of distances from two foci rather than the sum of the ellipse; The Cassini parametric form is

$$\sqrt{\frac{2a^2\cos(2t)+2\sqrt{b^4+(\cos^2(2t)-1)a^4}}{2}}\cos t \tag{6}$$

The ellipse can be generalized to the n-ellipse with more than two foci, allowing egg like shapes, though I've found no easily applicable parametric solution yet in the literature. However, there are other ways of making egg curves [14], including this simple formula attributed to Torsten Sillke: $y^2 = |\sin x + 0.1\sin 2x|$ easily applied in synthesis.

Two other more exotic curve generators are now introduced. The super toroid is a 3-dimensional surface related to the superellipse, whose x co-ordinate has the parametric representation:

$$\cos^a(\theta)(r_0 + r_1\cos^b(\phi)) \tag{7}$$

There are six controllable parameters in this case. Figure 3 demonstrates output where all six are being slowly randomly varied, exploring the output space of the algorithm.

Another more radical parametric form, further generalizing the superellipse, is the 'Superformula' [15]. [3] The

[1] In general, the reader may wish to investigate T-norms and T-conorms, see http://en.wikipedia.org/wiki/T-norm

[2] This was discussed on the sc-users mailing list https://www.listarc.bham.ac.uk/lists/sc-users/msg09524.html; in particular, James McCartney suggested adding a further term inside the atan, representing a constant addition that acts as a slant to the y axis of the resulting waveform.

[3] There are some excellent Processing examples of using this to synthesize visuals, see for example http://chamicewicz.com/p5/superformula3d/

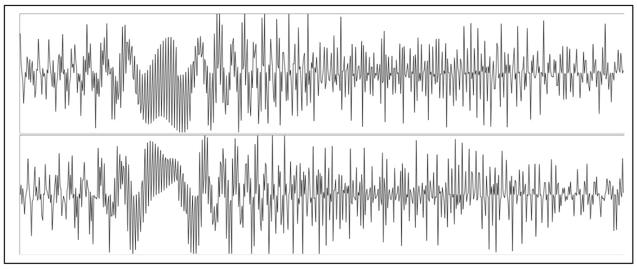

Figure 3. Run over 0.5 seconds of a super-toroid based oscillator (x and y co-ordinates as left and right channels) with control parameters swapped at 30Hz (15 times over this plot): $0.15 < r_0 < 0.35 \quad 0.4 < r_1 < 1.0 \quad 0.04 < a, b < 3.96$ and permissible frequencies from 5 to 600Hz

equation for the x co-ordinate in terms of polar angle θ:

$$(|a\cos(0.25m\theta)|^{n_0} + |b\sin(0.25m\theta)|^{n_1})^{\frac{-1}{n_3}} \cos(\theta) \quad (8)$$

reveals 6 controllable parameters; the cos and sin could also be decoupled with different phase updates.

The famous curves index at `http://turnbull.mcs.st-and.ac.uk/history/Curves/Curves.html` may provide the reader with plenty of further ideas.

6. QUATERNION MUSIC GENERATION

Hamiltonian numbers have application in computer graphics, for efficient calculation of 3D transformations. They can be applied in sound synthesis where the Hamiltonian product (quaternion multiplication) leads to some interesting number sequences. Starting from four dimensional quaternion points a_1 and b of unit norm form

$$a_n = a_{n-1} \otimes b \quad (9)$$

where $\otimes$ denotes quaternion multiplication. a_n has four components, providing up to four output signals. b is used over and over as the second term of the product, whilst the first term is updated. The amplitude will be damped over time if the norm of b is less than 1, and the amplitude can explode if larger than 1. As long as b is kept normalized as required, its components can be modulated over time; if b's magnitude drops, damping down a_n, a trigger could reset b.

This operation can also be nested, such that output at stage n, a_n, is itself used as b in another calculation. A more complicated network of quaternion oscillators can be established, possibly with feed back via quaternion addition (which is equivalent to vector addition of the four element vector representation).

The basic algorithm provides quite effective pitch modulated granular sounds; an iterative system with a little feedback takes on more complex rhythmic skittering. Figure 4 illustrates the later with time domain and spectrographic plots. More complicated quaternion nonlinear filters may lead to interesting further sounds.

7. NONLINEAR FILTERS MODULO 1.0

Consider the general equation:

$$y_n = ((\sum_{j\in J} c_j \prod_{i\in I_j} y_{n-i}^{\alpha_{ij}}) + x_n)\%1.0 \quad (10)$$

which takes a nonlinear filter of previous outputs, and a forcing input x_n which can used to push energy into the system as desired.[4] J and the I_j for each crossterm are indexing sets; most combinations of y_k will have constant zero, and the indexing sets let us write a system in a sparse rather than combinatorially explosive fashion.

The modulo 1.0 is absolutely critical, since the system will quickly blow-up in output values without it; the topological restriction to the interval $[0, 1)$ guarantees safety of output.[5]. The recurrence relation system here takes in some one dimensional discrete chaotic systems, such as the dyadic map (multiple simultaneous equations and more advanced crossterms would be required to cover more types).[6]

The modulo operation in a recurrence relation is also reminiscent of linear congruential generators $y_n = (a * y_{n-1} + c)\%1.0$, which have been used to directly synthesize sound, and as number sequence generators [6]. For musical applications, choices of parameters which would be poor for creating statistically effective pseudo-random number sequences are often more interesting, creating waveforms with noticeably short periods rather than (near) white noise; the current advice in computer science to never use LCGs for random number generation can be productively ignored by musicians seeking periodic waveform resources [17, p. 342].[7] Similarly, the general equation above can lead to periodic sounds (as limit cycles of chaotic equations), though there is also a danger of hitting fixed points; the x_n term can be used as a way of injecting fresh energy into the system as needed (the y_k memory entries can also be reset or re-initialised to new values).

Figure 5 is a time domain and spectral plot of the output of the specific equation:

$$y_n = (c_0 \prod_{i\in 1..10} y_{n-i} + c_1 y_{n-10} y_{n-7} + c_2 y_{n-9} y_{n-8} + c_3 y_{n-6})\%1.0 \quad (11)$$

created through this method, synthesized with random new y values and constants around once every 2000 samples. Moments of periodicity are observable, as well as zero DC (fixed points)[8] and more noisy passages.

[4] The equation could be further generalized to include inputs as well as outputs in the crossterms.

[5] For an alternative topologically safe nonlinear oscillation principle, see [16]

[6] For a really complex system, the index set integers could themselves be changed over time, as per Hofstader recursive sequences like the meta-Fibonnaci, even modulo an available memory length.

[7] However, whilst there are many non-LCG random number generation algorithms in computer science which have not been used explicitly in computer music as generators [17, chapter 7], experiments with Marsaglia's multiply-with-carry (ibid, pp. 347) and Knuth's subtractive generator (ibid, p. 354) proved them to be too robust in their noise characteristics to be musically interesting; this author hasn't yet managed to get anything but white noise-like sounds from them!

[8] Any zero output propagates through the delay line of past outputs, cancelling certain product terms, and may contribute to dampening down to a zero fixed point. Other DC offset values are possible as fixed point solutions too, in general.

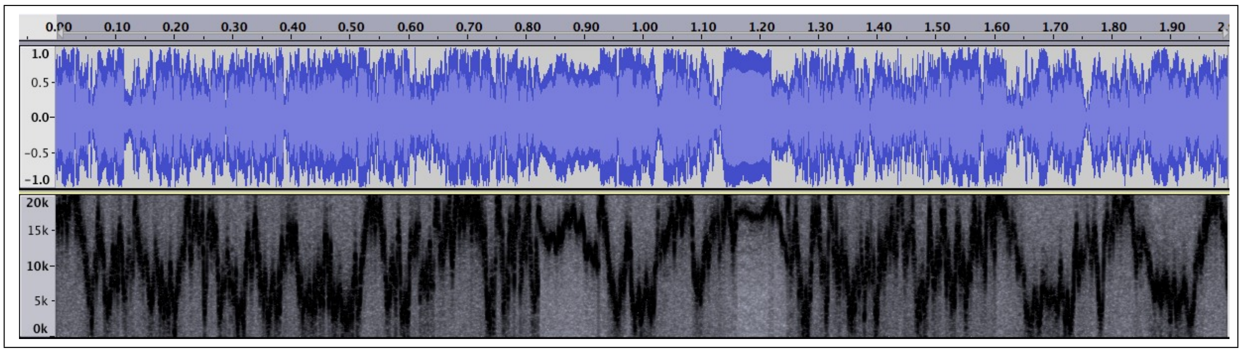

Figure 4. 2 seconds of output from a quaternion oscillator feedback system (three stages). The FFT for the spectrogram in the lower plot is a 512 point transform with Blackman window.

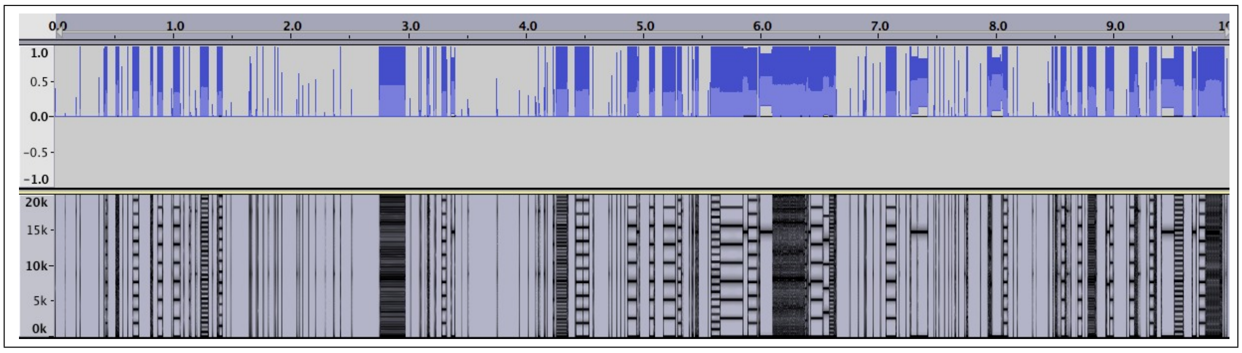

Figure 5. 10 seconds of output from a nonlinear modulo 1 filter (equation 11); the time domain waveform is left unipolar here in the generated range 0.0 to 1.0. The FFT for the spectrogram in the lower plot is a 512 point transform with Blackman window.

8. ALTERNATIVE CONCATENATIVE SYNTHESIS

Concatenative sound synthesis [18, 19] allows the repurposing of existing sound data, by labelling windows of samples by more sonically and musically meaningful summary feature values. A large audio database, so tagged, can be hunted for matches to an input sound, by some criteria of similarity and continuity of access. Two further variations on concatenative synthesis are now presented.

8.1 Sparse concatenation

Sparse approximation is an analysis technique which seeks out a representation for an input sound by selecting a subset of primary descriptive atoms from a larger dictionary of possibilities [20]. A greedy algorithm, matching pursuit (which has a number of variants) is the standard way to identify exactly which atoms are the most descriptive of the analyzed sound. Software like MPTK (Matching Pursuit ToolKit) allows one route to investigate this.

Following a study into cross-synthesis and morphing of sounds through dictionary-based methods [21] custom software was created to embody those techniques. It was straight forward to adapt that software to apply matching pursuit decomposition of a target sound, using a dictionary constructed of small windows of signal (at various resolutions) from a source. The similarity function (maximal inner product score) is the natural one for sparse approximation.

Reproductions of target sounds via this technique improve in accuracy with the number of iterations of matching pursuit, though perfect reconstruction is not theoretically possible (the source is not necessarily a union of bases of function space), nor indeed the musical goal. Distortions in the recreation of the target via the source's atoms have a flavour of the spectral and short-term temporal (the atoms are limited by window size) content of the source; the residual is also interesting, in the case of imperfect reconstruction. The aural result is like a rough-edged version of the original, with echoes of the source atom database pushing through the signal. Although more sophisticated analysis-resynthesis processes on the atoms themselves in reconstruction is an area of current research in concatenative synthesis [22], this technique may provide a further compositional resource, where often imperfect matches are of greater note.

8.2 EEG-led concatenation

Rather than direct audio to audio matching, audio can be tagged by derived feature values, even by auxiliary data, such as the current state of a controller causing the audio production. In this variant, EEG signal channels are used as the tag for music being listened to by the experimental subject. The idea is then that on thinking of the music or variations on it, new EEG signals can be used to access the tagged database, allowing brain to sound concatenative synthesis control. Brain to sound interfaces are receiving some attention at present as musical controllers [23, 24]; we used a standard (low-grade, 1 channel) commercial interface, the Neurosky Mindwave. One channel of raw EEG data was collected while listening to Beethoven's eighth symphony, last movement; the sample rate was 100 Hz, and a vector of the last 50 values was formed to tag half second chunks of audio (that playing back at the same time as the EEG data was collected). In a playback mode, the subject then mentally imaged parts of the Beethoven, in the naive hope that this would provide similar EEG trails and recover similar parts of the database.

The longer-term principle of this synthesis method could be profound. Unfortunately, on the small scale brain surface EEG we've been able to test with so far, no significantly controllable results have yet been delivered. Nonetheless, the application was fun to explore, if rather random in resultant. A higher research grade EEG with more channels may provide better traction.

9. CONCLUSIONS

This paper has presented a range of alternative sound synthesis methods. Some may prove productive for musicians and sound artists, and worthy of further investigation. Some may prove evolutionary deadends in sound space, but the spirit of exploration should not be inhibited by a few false starts. Most of these techniques await more formal mathematical analysis of their properties; this does not stop us from exploring them in practical application. Combinations, such as nonlinear filters modulo 1.0 applied in complex number or quaternion space, abound with further potential for radical computer music. Code and sound examples to accompany this paper are made available at http://www.sussex.ac.uk/Users/nc81/evenmoreerrant.html.

Acknowledgments

With thanks for discussion and insight to Peter Hoffman and Chris Ariza on Xenakisian sieves, and Peter Giesl for discussion of chaotic equations. Scott Wilson first pointed me to the hair style/oscillator paper, and Chris Kiefer, Matthew Yee-King and Mick Grierson were helpful in making EEG equipment available for tests. An sc-users mailing list discussion of the parametric ellipse equation involved Julian Rohrhuber, James McCartney, and Jonatan Liljedahl.

10. REFERENCES

[1] N. Collins, "Errant sound synthesis," in *Proceedings of the International Computer Music Conference (ICMC)*, Belfast, August 2008.

[2] J. O. Smith, "Viewpoints on the history of digital synthesis," in *Proceedings of the International Computer Music Conference (ICMC)*, 1991, pp. 1–10. [Online]. Available: http://ccrma.stanford.edu/~jos/kna/kna.pdf

[3] X. Serra, "Current perspectives in the digital synthesis of musical sounds," *Formats*, vol. 1, 1997. [Online]. Available: http://www.iua.upf.es/formats/formats1/a07at.htm

[4] M. V. Mathews, *The Technology of Computer Music*. Cambridge, MA: MIT Press, 1969.

[5] Viznut, "Algorithmic symphonies from one line of code – how and why?" 2011. [Online]. Available: http://countercomplex.blogspot.com/2011/10/algorithmic-symphonies-from-one-line-of.html

[6] C. Ames, "A catalog of sequence generators: Accounting for proximity, pattern, exclusion, balance and/or randomness," *Leonardo Music Journal*, vol. 2, no. 1, pp. 55–72, 1992.

[7] G. Strang, "Wavelets and dilation equations: A brief introduction," *SIAM Review*, vol. 31, no. 4, pp. 614–627, 1989.

[8] S. Hollis, "A brief history of oscillators and hair styles of European men," in *AASU Math/CS Colloquium*, 2002. [Online]. Available: http://countercomplex.blogspot.com/2011/10/algorithmic-symphonies-from-one-line-of.html

[9] P. R. Cook, *Real Sound Synthesis for Interactive Applications*. Wellesley, MA: AK Peters, 2002.

[10] P. Hoffman, "Xenakis alive! extensions and ramifications to xenakis' electroacoustic legacy today," in *Proceedings of the Xenakis International Symposium*, London, 2011.

[11] C. Ariza, "Sonifying sieves: Synthesis and signal processing applications of the xenakis sieve with python and csound," in *Proceedings of the International Computer Music Conference*, Montreal, 2009.

[12] S. Tipei, "Solving specific compositional problems with mp1," in *Proceedings of the International Computer Music Conference (ICMC)*, Texas, 1981.

[13] E. Weisstein, "Ellipse." [Online]. Available: http://mathworld.wolfram.com/Ellipse.html

[14] J. Köller, "Ovals and egg curves." [Online]. Available: http://www.mathematische-basteleien.de/eggcurves.htm

[15] J. Gielis, "A generic geometric transformation that unifies a wide range of natural and abstract shapes," *American Journal of Botany*, vol. 90, no. 3, pp. 333–338, 2003.

[16] G. Essl, "Circle maps as a simple oscillators for complex behavior: I. basics," in *Proceedings of the International Computer Music Conference (ICMC)*, New Orleans, 2006.

[17] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes: The Art of Scientific Computing (3rd ed.)*. New York: Cambridge University Press, 2007.

[18] D. Schwarz, "Data-driven concatenative sound synthesis," Ph.D. dissertation, Université Paris 6, 2004. [Online]. Available: http://recherche.ircam.fr/equipes/analyse-synthese/schwarz/

[19] B. L. Sturm, "Adaptive concatenative sound synthesis and its application to micromontage composition," *Computer Music Journal*, vol. 30, no. 4, pp. 46–66, 2006.

[20] S. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*, 3rd ed. Amsterdam, The Netherlands: Academic Press, Elsevier, 2009.

[21] N. Collins and B. Sturm, "Sound cross-synthesis and morphing using dictionary-based methods," in *Proceedings of the International Computer Music Conference (ICMC)*, Huddersfield, 2011.

[22] G. Coleman, E. Maestre, and J. Bonada, "Augmenting sound mosaicing with descriptor-driven transformations," in *International Conference on Digital Audio Effects (DAFx)*, Graz, Austria, 2010.

[23] E. R. Miranda and M. M. Wanderley, *New Digital Musical Instruments: Control and Interaction Beyond the Keyboard*. Middleton, WI: A-R Editions, Inc., 2006.

[24] M. Grierson, C. Kiefer, and M. Yee-King, "Progress report on the EAVI BCI toolkit for music: Musical applications of algorithms for use with consumer brain computer interfaces," in *Proceedings of the International Computer Music Conference (ICMC)*, Huddersfield, 2011.