8. **DODGE,C. and JERSE,T.A.** (1985), Computer Music, Schirmer Books N.Y. .

9. **DUFOURT, Hugues** (1981)- "Les difficultés d'une prise de conscience théorique", in Le compositeur et L'ordinateur, IRCAM, Paris.

10. **GRISEY, G.** (1989), "Tempus Ex Machina", in Entretemps n° 8, Paris, France.

11. **LORRAIN, D.** (1980), Une Panoplie de Canons Stochastiques, Rapport IRCAM-n° 30, Paris.

12. **MALT, Mikhail** (1991)- Trois aspect de formalisation dans Achorrispsis de Iannis Xenakis , Mémoire de D.E.A.,sob a orientação de Huges Dufourt, EHSS-Ecole des Hautes Études en Sciences Sociales et IRCAM-Institut de Recherches et Coordination Acoustique Musique.

13. **MALT, Mikhail** (1992)- PW-Alea-librairie de Modèles stocastiques, IRCAM, Paris, France.

14. **MALT, Mikhail** (1993)- Introduction à Patchwork , IRCAM, Paris, France.

15. **MALT, Mikhail** (1994)- Chaos-librairie de modèles chaotiques et de fractales , IRCAM, Paris, France.

16. **Mc ADAMS, Steve et A. Bregman** (1987)- "L'audition des flux musicaux", in Marsyas, Institut de pédagogie musicale et chorèographique, La Villette, Paris (3-4) décembre 1987, PP 97-118.

17. **MURAIL T.** (1989) - "Questions de cible " in Entretemps n° 8, Paris, France.

18. **PEITGEN, H. O. ; JÜRGENS, Harmut; SAUPE, Dietmar** (1993)- Chaos and Fractals, New Frontiers of Science, Springer-Verlag, New York.

19. **WALLISER, Bernard** (1977)- Systèles et Modèles, Editions du Seuil, Paris.

20. **XENAKIS, I.** (1976), Musique Architecture, Casterman, Paris

21. **XENAKIS, Iannis** (1981) - Musiques Formelles, Stock Musique, Paris.

7.Agradecimentos

Escrevo este artigo como como bolsista do CNPq na EHSS-IRCAM-Paris, para a realização de um Doutorado em Música e musicologia do século XX sobre "Modelos Matemáticos e Composição Assistida por Computador" sob a direção de Huges Dufourt e Jean Baptiste Barrière.

Synthesizing Music with Sampled Sound

YeeOn Lo (muse@leland.stanford.edu) and Dan Hitt (hitt@cs.stanford.edu)
Center for Computer Research in Music and Acoustics
Stanford University
Stanford, California 94305
USA

Abstract

Sampled sounds are now an important resource for modern music-making and multi-media events. But except for certain classes of sampled sounds, digital synthesis methods have to be crafted to exploit them fully. A method involving a certain form of analysis is described here. Some results are presented and its musical applications discussed.

1 Emergence of Sampled Sounds

Sampled sounds are now an important resource for modern music-making and multi-media events. As magnetic storage costs less than $1 (US) per megabyte and gets cheaper all the time and as flash memory becomes increasingly available, the advantage of sampled sounds in music computing is obvious. For example, there is much less compelling reason to use an algorithm to generate, or synthesize, plucked-string sounds when the latter are readily available from a CD-ROM or hard-disk memory.

Using samples not only speeds up the run-time process, thus making real-time performance attainable after a certain threshold in hardware speed is crossed, but also saves the user development cost, because the operation becomes as simple as file I/O management instead of coding and debugging a piece of numerical calculation in a typically larger and more complex music computing environment (which combines and manipulates samples)[1]. In other words, the space advantage of algorithmic synthesis is now overshadowed by time considerations.

So it seems that samples are replacing synthesis at least where acoustic instrument timbres are concerned. And indeed one might argue that any sound of nonacoustic origin may be similarly made available as samples at the factory, saving user cpu as well as development time as discussed above[2].

2 Potential and Controversy

Now by means of widely available sequencing software, one can easily explore combining samples with control over choice of sound material, amplitude, timing, spatial movement and even reverberation. Therefore it is not surprising that some see samples to replace the orchestra soon if not already. Surprising, however, is that not everyone shares this bright outlook and there are those who are just as vehement in believing that even from a purely musical standpoint, sampled sounds (with all the help they can get from the computer) will never replace the orchestra!

Thus the questions are: Do we still need synthesis in a widely applicable sense? That is, do we still need synthesis if we are only interested in making music from available sampled sounds? If we do, why? And what form of synthesis do we need?

To answer these questions, it might be profitable to examine some of the issues pertaining to working with sampled sounds. Those who believe in the role of the orchestra or live acoustic ensemble more or less put their money where their ears are. Most trained musicians who rely on their ears to do their business will say the music written for an orchestra and realized by sampled sounds are "second-rate" at best. (So far we haven't examined the source of this less-than-second-ratedness.)

[1]To be sure, we might still want to filter the samples, detune them, shape them and "warp" them in all manner one can imagine, superpose them and sequence them, but these would be necessary additional operations anyway in the music synthesis paradigm whether samples are used or algorithm are invoked. The advantage of sampled sound in this instance illustrates the advantage of a whole class of sounds which are point-excited in origin—contributing to the popularity of drum machines.

[2]Here synthesis is used in the traditional sense of the word in sound synthesis by digital computer: the generation of a sequence of numbers that approximates the waveform to be heard (excluding a transformation for scaling—within the limits of linearity—which is nothing more than turning a dial on your amplifier in the analog domain).

On the other hand, those who are educated in the language of information theory and computer technology see a different perspective, based on a string of existential and counting arguments: Surely the Sampling Theorem of Nyquist and Shannon guarantees that every vibration in the air that our ears can catch the computer can duplicate with arbitrarily high fidelity. So if we sample as often as is required by the rate of the most stringent fluctuation in the acoustic signal and resolve each sample value into a number over the range required by the dynamic range of the signal and simultaneously minimize the noise from quantization (which, although it cannot be completely removed, is generally not a musical objection as far as concert music is concerned), then all the fine details in the orchestral performance will indeed be faithfully captured. Thus, at least at a single spatial point (or a finite collection of them), a piece of orchestral music is simply a finite sequence of numbers having finite resolution—i.e., equivalent to a finite sequence of integers. Bounding the length of the piece (say to require it to be less than one hour) bounds the number of such sequences, and gives a finite space of integer sequences to search through for any desired orchestral piece (of duration less than one hour).

It is, however, a leap of faith to suggest that a large enough cascade or assembly of oscillators will approximate arbitrarily closely the musical waveform if a few of them have been demonstrated to do so to some degree of success with a certain (sub-)class of sounds. In this case, there is no mathematical theorem concerning the procedure of *synthesis* (as opposed to counting). The Fourier theorem has stringent assumptions. When we stretch the domain of operation to suit these assumptions, i.e., take the Fourier Transform of the entire piece so as to avoid the periodicity restriction, we lose complete control over the procedure of manipulating the parts (the samples) that form the piece. This is a problem because our ears make use of time domain information as well as frequency domain information; in fact, music is traditionally written, performed, and listened to in the time domain. When we "violate" the Fourier premises by using time-varying approaches, we encounter all kinds of artifacts, as well as having to wrestle with tedious computations and grapple with precision of control. In short, there is *no* theorem that will *guarantee* the existence of a recipe which would produce a digital waveform for a score that would closely approximate the digitized copy of an orchestral rendition of that score.

3 A Fundamental Practical Constraint

So, the enumeration and existence arguments—that the set of all digital waveforms forms a superset of acoustic waveforms (and hence include every single orchestral piece ever written or that could be written or that are physically realizable)—ignore the intricate process where by the overall waveform is eventually arrived at. The enumeration arguments are simultaneously at odds with the combinatorial explosion problem. These arguments, after all, apply even more strongly to writing prose, for example, but authors have not been replaced. The moral is that a *finite* search space is not necessarily *small*, at least from a human perspective. On the other hand, we have a small alphabet from which to build a rich written vocabulary (and thus a literature). And we have a small set of phonemes to build a rich and flexible spoken vocabulary (and thus can communicate). So it is quite desirable, from a computer-science standpoint, to have a library of elemental objects (such as the sampled sounds) from which to build complex music from—if we ever hope to have a satisfactory solution to making music with sampled sounds.

It might be easy to dismiss our current lack of success in realizing an orchestral score via sampled sounds as a matter of incompatible paradigms—the methods of generating orchestral and computer music being too far apart. But one doubts this view satisfies most who are dedicated to maximizing the utility of a modern digital computer and who are well aware of the expressiveness and power possible from a computer (theoretically, and, in other areas, practically). The machine's current lack of eloquence reflects our own (current) limitations.

An alternative might be to explore the sources of obstacles that give rise to that second-rate quality when a score is realized with sampled sounds: identify the problems and search for solutions.

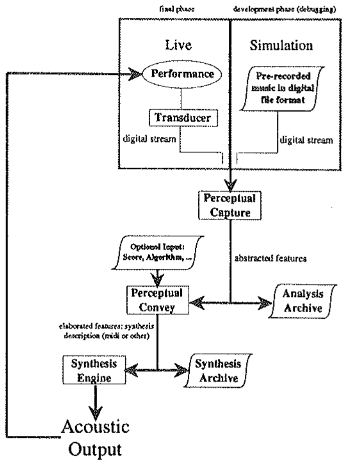
To do this, we will first take a step back to examine an important musical application of sampled sounds: interactive performance. Through it, we will soon find out some fundamental problems that hinders their utility for making superior-quality music. And then we will proceed to suggest solutions and present some preliminary results.

4 Interactive Performance: Can a machine imitate?

A classical approach to music making is the method of imitation: canon, ricercar, fugue, or almost any polyphonic writing, especially ensemble pieces.

Can a machine imitate? How well can it? Especially with sampled sounds.

Without loss of generality, let us use the model depicted in the diagram to discuss a situation in interactive performance—an important application of the computer to music. Let's suppose that the live part consists of a clarinet and a viola. Now suppose that the clarinet begins the music with a lyrical solo passage and we expect the machine to ease in with a polyphonic but homogeneous texture. (We are looking for a gentle build-up in the musical activity.) Such a development naturally calls for a similarly lyrical line from the machine, ideally voiced by a similar, i.e., clarinet, timbre. Now, if the lyricism of the live performer involves slurring of notes into long and short phrases as is most often expected, the polyphony would make the most sense if the machine counterpoint is also slurred.



We will not consider the method of generating the counterpoint here (which is by no means trivial, although it can certainly be achieved to varying degrees of success). The issue to discuss here is articulation. Those of us who have worked with sampled sounds are aware that machines don't slur very well. And that is because despite the fact that a wealth of information is transmitted from the musicianship of a skilled performer to the acoustic signal, all a machine (a commercial synthesizer or otherwise) can do is cross-fade, which conveys practically zero information. This is an obstacle not only to using sampled sound to jam with the clarinet, but also to jamming with the viola, a bowed-string instrument.

This ability to slur, or do other kinds of articulation over a group of notes, which contributes so much to the music of any acoustic instrument playing, and which is such a measure of musicianship of the performers, is lacking in current applications of sampled sounds with computer [1]. The lack which is indeed a major stumbling block to replacing the orchestra (with sampled sounds) is due to a fundamental limitation in our understanding of signal behavior when notes are joined, i.e., when samples of different pitches but similar timbres, or different pitches and different timbres, or even different dynamics, are joined. Although one can avoid the issue and create another kind of work, the limited machine expressivity means the available "tricks" may be exhausted before long and adequate jamming or imitating using such classically versatile techniques as transposing, inverting, retrograding, or permuting as Bach might have done, is not practicable. (The emphasis is on the word *adequately* here; a preprogrammed or even spontaneous note list whose performance is not musical or bears only a poor relation to the performance of the live player is not really adequate.) Hence until workable methods for slurring are found (which may in the worst case differ from sample pair to sample pair), it is safe to say that the computer will not replace the orchestra (nor will sampled sounds).

5 A New Kind of Synthesis

The slurring or transition problem cannot be solved by recording and storing tables of transitions; this runs into the "combinatorial explosion" problem so familiar from computer science. And we still have the problem of joining the steady-state sampled regions with the transitions. This means we have to solve the problem of joining sampled sounds, and points to the need for a more sophisticated synthesis than we have commonly available.

Here, by synthesis, we mean the generation of the sequence of numbers which describes a waveform or a section of a waveform—such as the transitions between two samples. In this case, the samples near the transitions may also be modified in order that the connection be smooth.

In order for the synthesis to be capable of creating smooth transitions between sampled events, so as to achieve greater intimacy of interaction with live performers and to possess a greater degree of coherency, analysis must play an important role in this kind of synthesis. For example, if we need to create a transition between clarinet tones α and β, we must somehow dig into the data present in α and β, and elicit its essence.

The question now shifts to *what* kind of analysis is suitable for analysis-based synthesis.

6 Perceptual Computing, and "Capture and Convey"

Analysis has value in and of itself; it is, after all, the primary intellectual activity.

The analysis that we propose should be performed on sounds, however, has a particular goal: to enable a suitably flexible synthesis for musical purposes.

This suggests that our analysis should be "receiver-based": i.e., it should seek out attributes of the signal which receivers (humans) find significant. Further, it should guarantee some sort of continuity or smoothness condition in the other (auditory) perceptual attributes as a particular one is varied, in the synthesis of a class of sounds. In short, the analysis of a particular perceptual attribute should be able to express its essentials in the context of other perceptual attributes. For example, a computation which helps to enable us to take the pitch contour of a phrase with one timbre and re-perform it with another timbre would meet this criterion. A computation which purports to abstract a pitch contour but from which we cannot develop or reconstruct a phrase (e.g., to make a satisfactory slur) would not meet this criterion, and in fact would not be nearly as useful. A second example would be a process enabling one to take a pitch contour in a given timbre and transpose it to another pitch height without significantly distorting the timbre—we might refer to this as **capture and convey**, and we maintain that we haven't really captured anything if we cannot convey it (i.e., in this case, transpose it without distortion) We can formalize conveyability and how we intend to use it:

Definition. A perceptual structure σ is *conveyable* (χ) (over a sound set S with respect to pitch) if

$$||P_\sigma(x) - P_\sigma(x')|| \ll ||P_\pi(x) - P_\pi(x')|| \quad x, x' \in S$$

Here, P_π is the projection onto the pitch dimension, i.e., $P_\pi(x)$ is the pitch of x, and P_σ is the projection onto the parameter space of σ.

Remark. We cast conveyability in these terms because often changing pitch necessarily changes σ (which might be one of the "qualities" of timbre, for example, to what extent and in what way it sounds like a violin); we only demand that the change in σ be small compared to the change in pitch.

Definition. A perceptual structure σ is *capturable* (κ) weakly for a certain sound if that sound can be resynthesized perceptually identically with respect to σ. It is capturable strongly if as well it is conveyable.

Remark. The intention is that we've captured a particular timbre weakly if we can resynthesize it, and we've captured it strongly if we can resynthesize it along some range of pitches.

These considerations—in particular the notion of capture and convey—provide us with a broad measure of the success of our analysis and synthesis as well as measuring how resonant our methods are with sampled sounds.

7 A Particular Case: the Kinematic Method

Here, we report on a particular kind of analysis-based synthesis which we are using and trying to further develop to deal with the issues mentioned previously: joining sound events, capture and convey, etc.

We call the method **kinematic synthesis** because it is modeled on certain entities moving (hence "kinematic") through a suitable vector space (see [2], [3], and [6] for additional information on the kinematic method).

Briefly, the method models musical sounds as consisting of states and transitions between them, where the time scale of a state is comparable in some cases to the time scale of frames in a motion picture; in fact, we call the "periods" of a sound "frames". We note of course that no musically interesting sound is periodic, but the waveform of every musically interesting sound passes through stages with a great deal of redundancy involved. One goal of kinematic synthesis is to remove the redundancy from the analysis data—but not remove the "content".

More exactly, the method postulates that a sound can be perceptually recaptured from a certain set of analysis data, a **triple**, consisting of a **frame trajectory**, an **amplitude envelope**, and a **period trajectory**. The use of the term "trajectory" is to suggest motion. The frame trajectory does not consist of all the frames (as we've used the term above) in the sound, but principle or key frames (**break frames**, by analogy with break points in a piecewise linear or piecewise smooth curve) from which much of the sound can be recovered (some might call it spectral data, which is approximately true). The amplitude envelope maps out the overall changes in amplitude, and can be cast in many forms (including as a pair of functions: a lower and upper envelope). There is actually a great deal to be said about the amplitude envelope (as there is about the other two members of a triple), but one heuristic guide to its construction is that it reflect the smoothness and evolution of the sound. The period trajectory maps how the "micropitch" varies (and in many cases is nearly flat) as measured by the changes in the motion of the frames through the space.

The analysis data (**triples**—frame trajectories, period trajectories, and amplitude envelopes) is directly interpretable from a perceptual view.

The method tries to exploit the mass of sampled data we find ourselves surrounded with, but tries to cut redundancy on a dynamic basis (so in a very "innovative" part of a sound, say the attack, more frames—more data—will be used, as well perhaps having a more densely specified amplitude envelope). If we cast the waveform as a series of frames—points in some high-dimensional vector space—it forms a curve, which we sample more densely where it is most dynamic (e.g., where the curvature is greater). Our data reduction is typically in excess of 90% just on resynthesis (capture) using the break frame notion—and we lose no phase spectrum data (on the break frames).

The method has had some success in capture and convey; we've conveyed flute tones across an octave, and a violin tone across three octaves with variable (and arbitrary) durations. We've also conveyed a slurred two-note phrase from the flute in the pitch dimension, and have created timbre melodies involving the violin and the flute. If we convey a sound over M pitches, then of course the data reduction is M times as large.

8 Applications

Kinematic synthesis, or other comparable methods, have at least four important uses in musical constructions involving sampled sounds:

1. Transposition, retrogression, inversion, and other continuous line formations. The first three of these are self explanatory (and of course, when we talk of, say, inversion, inverting the note list is the tiniest part of the effect we want to produce: we want to invert a line with articulation and phrasing suitably related to the original). By "other continuous line formations" we refer to effects on continuously changing sounds, such as slowing or other articulation.

2. Timbre substitution. By this we mean taking a line and holding its pitch trajectory constant while varying other components of the timbre (in the simplest case, this might refer to say, replacing a clarinet line with a trumpet line—musically).

3. Analogically reasoned transformations. In a simple case, this might refer to going through a curve, seeking a feature sequence $\alpha\beta$ where α stands in relation to some feature A in a template in the same way the β stands in relation to feature B in the template, and then replacing β by a τ (from a T in the template). See [3] regarding this approach to sound modification-synthesis.

4. Restoration of data from corruption. A very good analysis-based synthesis paradigm should be able to go through corrupted data—say captured from an old vinyl record—and create a digital stream *perceptually* identical to the original. (We don't claim to have done this on any large scale.)

The greatest interest one has in these applications is of course in the case of continuously sustained sounds. An ideal computing environment to carry out this kind of analysis-based synthesis is an open-ended (extensible) object model with a "polyhedral" architecture where multi-window, multi-document applications form the vertices of the polyhedron, and can communicate with each other across the edges, under user control; mixing, editing, synthesis, and analysis can be done concurrently in full view of the user ([5], [6]).

9 Concluding Remarks

1. Certain classes of sampled sounds are directly, immediately, and readily usable for the purposes of composition, e.g., point- (impulse-) excited sounds such as plucked string (pizzicato), drum, and piano (to some extent). Other sounds are useful for special occurrences but not for smooth melodic definition.
2. In order to widen the applicability of sampled sounds, the research community needs to direct its efforts to finding solutions to make the non-impulse-excited sounds readily usable for forming musical constructs. A synthesis model is not as useful if its demonstrable application is limited to plucked or percussive sounds. Likewise, a synthesis model which can take two sampled sounds and join them in a variety of expressive two-note phrases is more useful than a model which merely duplicates an existing sampled sound.
3. We must recognize that making a lyrical melody via digital means entails more than just cross-fading two sounds. It is essential to accept that in most cases, creating transitions requires knowledge, either in the form of data from the signal, or in equivalent algorithms to generate the data.
4. There is more than one possible transition between two tones, whether the timbres are the same or different. That is, a multiplicity of possible trajectories exists. These constitute a repertoire of articulations. The ability to display them digitally is a demonstrates the expressivity of a given composition/performance environment.
5. To choose a suitable transition trajectory (in order to maximize some local coherence criterion in composition), one needs suitable analytical tools. This is important in both a composition as well as a performance environment. The relevant tools might include some to decide which transition trajectory is being executed by the performer and provide the responding algorithm the best or most accurate information (regarding what actually happened) and allow it to make the best choice under a given composition strategy. They might also include some to analyze the pitch trajectory so as to perform a retrograde, transposition, inversion, etc., in a musically responsive way.
6. One synthesis candidate that is naturally suited for joining (or connecting) sampled sounds into articulated melodic constructs is kinematic synthesis, where the basic unit of the method is the triple: the amplitude envelope (possibly expressed as upper and lower sub-envelopes), the pitch trajectory or equivalent, and the trajectory of critical frames (which are periods in a steady state, and something more general in transient regions).

References

[1] Strawn, J. (1985) *Modeling Musical Transitions* (Ph.D. Thesis), Technical Report STAN-M-26.

[2] Lo, Y. (1986) "A Technique for Timbre Interpolation" *Proceedings of the ICMC* 241-248.

[3] Lo, Y. (1987) *Toward a Theory of Timbre* (Ph.D. Thesis), Technical Report STAN-M-42.

[4] Hitt, D. & Lo, Y. (1990) "*L*: A Language for Composition", *Proceedings of the ICMC* 237-240.

[5] Hitt, D & Lo, Y. (1992) "An Alternative Digital Environment for Music Synthesis" *Proceedings* of the Delphi Computer Music Conference/Festival [not paginated].

[6] Lo, Y. & Hitt, D. (1992) "Uniform Treatment of Sounds and their Syntheses on Digital Computers" *1992 International Workshop on Models and Representations of Musical Signals, Capri, Italy* [not paginated].

Um Ambiente de Auxílio a Composição Musical

ALEXANDRE JONATAN BERTOLI MARTINS
ANDRÉ LUIZ COSTA BALLISTA
Instituto de Informática - CPGCC
Universidade Federal do Rio Grande do Sul
Campus do Vale - Bloco IV
Av. Bento Gonçalves, 15064, CEP 91501 - 970
Porto Alegre - Rio Grande do Sul
fone: +55 (051) 336-8399 - Ramal 6161
e-mail: NATAN@INF.UFRGS.BR e BALLISTA@INF.UFRGS.BR

MARCELO SOARES PIMENTA
Departamento de Informática e Estatística
Universidade Federal de Santa Catariana
Campus Universitário, Trindade, CEP 88049-900
Florianópolis - Santa Catarina
fone: +55(0482) 319739
e-mail: CEC1MSP@BRUFSC.BITNET

Abstract:

Este artigo apresenta um ambiente de auxílio a composição musical desenvolvido em Smalltalk V/286. Ele provê ao compositor meios de criar e manipular objetos sonoros, os quais podem ser testados interativamente através da utilização de sintetizadores acoplados ao ambiente. Para este sistema não há distinção entre uma nota musical e uma melodia completa: ambos são objetos sonoros e são tratados da mesma forma. O sistema foi projetado e implementado de forma orientada a objetos e o ambiente utiliza o mesmo paradigma para sua interação com o usuário.

1. Introdução

As aplicações músico-computacionais têm se destinado a atividades bastante diversas. Aquisição e reprodução de performances, ensino e treinamento prático e teórico da música, geração automática de melodias e auxílio a composição musical são alguns exemplos.

Neste artigo, trataremos especificamente da utilização de computadores na composição musical. Como compor é um processo criativo, é necessário um ambiente que proporcione ao usuário ferramentas que o auxiliem durante todo este processo. Por sua vez, estas ferramentas devem ser projetadas especificamente para o processo de composição, de forma que o compositor não seja obrigado a adotar uma metodologia de criação para que possa utilizar tais ferramentas.

Apresentaremos aqui um ambiente de auxílio a composição musical denominado CAMC [2], que é um ambiente orientado a objetos desenvolvido em Smalltalk V/286 para suportar o processo de composição musical. Ele provê ao compositor meios de criar e manipular trechos musicais, os quais podem ser testados interativamente através da utilização de sintetizadores acoplados ao ambiente. Para este sistema não há distinção entre uma nota musical e uma melodia completa: ambos são objetos sonoros e são manipulados da mesma forma.

Para a representação da música utilizamos o modelo Smallmusic [3], que incorpora conceitos musicais ao ambiente de programação Smalltalk [5]. Todos estes conceitos são automaticamente mapeados para objetos da linguagem Smalltalk que podem ser manipulados e reutilizados na implementação de novas ferramentas computacionais voltadas para a música.