XVIII Congresso Nacional da Sociedade Brasileira de Computação "Rumo à Sociedade do Conhecimento"

ANAIS - Volume III
V Simpósio Brasileiro de Computação e Música

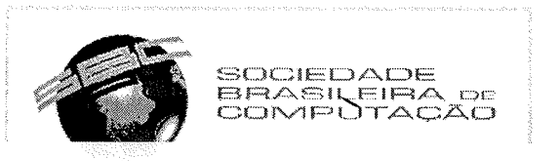
Campus Pampulha da UFMG
Belo Horizonte, Brasil
3 a 5 de agosto de 1998

Edição
Maurício Alves Loureiro (Escola de Música - UFMG)

Organização
Departamento de Ciência da Computação da UFMG
Escola de Música da UFMG

Elissandra Barbosa

Promoção



Impresso na gráfica da UFMG em julho/1998
Coordenação Editorial: João Paulo Kitajima (UFMG)
Arte gráfica do evento/capa: Zeca Campos

V Simpósio Brasileiro de Computação e Música (17.:1998 ago. 3 - ago. 5 : Belo Horizonte)

Anais/ V Simpósio Brasileiro de Computação e Música; editado por Maurício Alves Loureiro. - Belo Horizonte: Sociedade Brasileira de Computação, 1998.
xviii, 294 p.: il.

Evento realizado no XVIII Congresso da Sociedade Brasileira de Computação.

1. Computação e Música. I. Loureiro, Maurício Alves. II Congresso Nacional da Sociedade Brasileira de Computação (18.:1998 ago. 3 - ago. 5 : Belo Horizonte). III. Título.

Prefácio

Em 1994 foi criado um núcleo nacional de Computação e Música, NUCOM, com o objetivo de fomentar as atividades desta área de conhecimento no Brasil. No mesmo ano, o NUCOM realizou o *I Simpósio Brasileiro de Computação e Música*, em Caxambu, Minas Gerais, como um dos eventos integrantes do *XIV Congresso Anual da Sociedade Brasileira de Computação*. O evento contou com a participação de pesquisadores e artistas vindos da América do Norte, América Latina, Europa e Ásia e apresentou um total de 34 trabalhos científicos, 48 composições de música computacional, 5 conferências e 2 mesas de debates que reuniram pesquisadores brasileiros e estrangeiros em torno de temas relacionados à pesquisa e à formação universitária. O simpósio atraiu a atenção da comunidade internacional de Computação Musical e consolidou o Brasil como um dos maiores representantes da pesquisa nesta área na América Latina.

A partir de então, o NUCOM tornou-se uma Comissão Especial da SBC: *Comissão Especial de Computação e Música*, criada oficialmente no ano segunite em Canela, RGS, durante a segunda versão do simpósio, que como mais outras duas versões realizadas nos anos subseqüentes, respectivamente em Recife e Brasília, integrou um Congresso Anual da SBC.

Mais uma vez, o *V Simpósio Brasileiro de Computação e Música*, mostra o mesmo volume e a mesma qualidade de produção dos simpósios anteriores, sendo pesquisadores e artistas brasileiros responsáveis por aproximadamente a metade dos trabalhos selecionados.

Reafirma-se a consolidação desta área científica no Brasil, que tem seu abrigo institucional na : *Comissão Especial de Computação e Música* da Sociedade Brasileira de Computação. Com criação desta comissão especial, e o compromisso assumido perante seus objetivos, a SBC dá um passo institucional decisivo na direção de uma real associação entre arte e ciência, uma postura ousada, que não encontra muitos outros exemplos no mundo.

Gostaria de creditar esta ousadia aos membros da SBC que sempre apoiaram a *Comissão Especial de Computação e Música*, os quais faço aqui representar pelos nomes que estiveram mais envolvido neste processo: o Presidente da SBC, gestão até 1997, Ricardo Augusto da Luz Reis, o atual Presidente da SBC, Sílvio Romero L. Meira, assim como os presidentes dos Congresso Anuais da SBC desde 1994, Nívio Ziviani (UFMG, 94), Flávio Rech Wagner (UFRGS, 95), Rafael Dueire (UFPE, 96), Maria Elenita Menezes Nascimento (UnB, 97) e João Paulo Kitajima (UFMG, 98).

Belo Horizonte, agosto de 1998
Maurício Alves Loureiro

Prefácio do Coordenador do CNSBC'98

É com grande satisfação que assinamos o prefácio dos anais do XVIII Congresso Nacional da Sociedade Brasileira de Computação. Este evento congrega um conjunto de simpósios, workshops e seminários, além de premiações de trabalhos científicos, todos na área da Ciência da Computação. É o maior evento acadêmico do país neste ano de 1998, quando a Sociedade Brasileira de Computação comemora os seus 20 anos. Procuramos marcar este evento como aquele onde houve o maior fluxo de *informação* entre todos os envolvidos deste acontecimento. Serão vocês que nos dirão se esta estratégia foi adequada ou não. Aproveitamos para agradecer aos participantes da SBC pelo apoio constante e confiança, aos coordenadores, comissões e avaliadores dos eventos do CNSBC'98, ao ENECOMP'98, ao DCC/UFMG (professores, alunos e funcionários), aos patrocinadores e à Elissandra Barbosa, sem esquecer aqueles que nos completam fora do trabalho. Uma menção saudosa aos Professores Lia Golendziner (Informática/UFRGS), José Mauro Volkmer Castilho (Informática/UFRGS) e Luiz Martins (Informática/PUC-Rio) pelo legado que nos deixaram.

Belo Horizonte, agosto de 1998

João Paulo Kitajima

CNSBC'98

XVIII Congresso Nacional da Sociedade Brasileira de Computação

Coordenador Geral

João Paulo Kitajima, UFMG

Secretaria Geral

Elissandra Barbosa

Comissão Coordenadora

- Antônio Loureiro, UFMG
- Autran Macêdo, UFU
- Carlos Ribas, PUC-Minas
- Claudionor Coelho, UFMG
- João Paulo Kitajima, UFMG (coordenador)
- Maria Elenita Nascimento, UnB
- Ruy Milidiú, PUC-Rio

Convidados de Honra

- Dr. Virgílio Almeida, UFMG, Brasil
- Dr. Edmund Clarke, Carnegie Mellon University, EUA
- Dr. Daniel Menascé, George Mason University, EUA

Suporte Organizacional

DCC/UFMG
Instituto de Informática da PUC-Minas

V Simpósio Brasileiro de Computação & Música (SBC&M)

Coordenador Geral

Maurício Loureiro, UFMG

Secretaria Geral

Marli de Lourdes da Silva Coura

Comissão de Programa - artigos

- Aluízio Arcela, LPE, UnB
- Axel Mulder, Simon Fraser University, Canadá
- Daniel Oppenheim, IBM T. J. Watson Research Center, EUA
- Didier Guigue, UFPB
- Edilson Ferneda, UFPB
- Eduardo Miranda, University of Glasgow, Reino Fábio Kon, University of Illinois Urbana-Champaign, EUA
- François Pachet, LIP6, Université Paris VI, França
- Furio Damiani, UNICAMP
- Geber Ramalho, UFPE (coordenador)
- Gerard Assayag, IRCAM, França
- Henkjan Honing, University of Amsterdam, Holanda
- Unido
- Márcio Brandão, University of Edinburgh, Reino Unido
- Richard Moore, University of California San Diego, EUA
- Xavier Serra, IUA-Pompeu Fabra University, Espanha

Comissão de Programa - composições

- Celso Aguiar, CCRMA, EUA
- Craig Harris, Leonardo Electronic Almanac, EUA
- Denis Smalley, City University, Reino Unido
- Jorge Antunes, UnB

- Sergio Freire, UFMG (coordenador)
- Sílvio Ferraz, PUC-SP

Comissão de Programa - tutoriais

- Manuel Rocha Iturbide
- Martin Alejandro Fumarola, LEIM, National University of Cordoba, Argentina
- Juan Reyes, Universidad de Los Andes, Colômbia
- Mihail Malt, IRCAM, France
- Robert Willey, Fundação Carlos Gomes-Belém (coordenador)

Convidados de Honra

- Aluízio Arcela, LPE, UnB
- Celso Aguiar, Stanford University, EUA
- Curtis Roads, University of California at Santa Barbara, EUA
- Denis Smalley, City University, Reino Unido
- Eduardo Miranda, University of Glasgow, Reino Unido

Sociedade Brasileira de Computação

Diretoria

Sílvio Romero L. Meira, UFPE - Presidente
Flávio Rech Wagner, UFRGS - Vice-Presidente

Clarindo Isaias P.S.E. Pádua, UFMG - Editora SBC
Cláudia Bauzer Medeiros, UNICAMP - Publicações
Eratóstenes Edson de Araújo, SOFTEX - Divulgação e Marketing
Fábio Q. B. Silva, UFPE - Finanças
Guilherme Horta Travassos, UFRJ - Administração
Iára Pereira Claudio, PUC-RS - Secretarias Regionais
João Paulo Kitajima, UFMG - Eventos e Comissões Especiais
José Carlos Maldonado, ICMC-USP - Educação
Tarcísio Pequeno, UFC - Planejamento e Programas Especiais

Conselho

Cláudio Kirner, UFSCar
Cláudio Leonardo Lucchesi, UNICAMP
Daniel Schwabe, PUC-Rio
Júlio César Sampaio do Prado Leite, PUC-Rio
Paulo César Masiero, USP São Carlos
Paulo Roberto Freire Cunha, UFPE
Ricardo Augusto da Luz Reis, UFRGS
Roberto da Silva Bigonha, UFMG
Sérgio Bampi, UFRGS
Siang Wun Song, USP

Membros Suplentes do Conselho

Itana Maria Gimenez, UEM
Maria Elenita do Nascimento, UnB
Ricardo Anido, UNICAMP
Rosa Maria Viccari, UFRGS
Therezinha Souza de Costa, PUC-Rio

Secretarias Regionais

Minas/Centro-Oeste: Maria Elenita Nascimento, UnB
Norte/Nordeste: Hermano Perrelli de Moura, UFPE
Rio de Janeiro: André Monat, UERJ
São Paulo: Neucimar Jerônimo Leite, UNICAMP
Sul: Raul Ceretta Nunes, UFSM

Comissões Especiais

Arquitetura de Computadores e PAD: Líria Matsumoto, USP
Banco de Dados: Cláudia Bauzer Medeiros, UNICAMP
Computação e Música: Geber Ramalho, UFPE
Comp. Gráfica e Proc. Imagens: Carla del Sasso Freitas, UFRGS
Concepção de Circuitos Integrados: Marcelo Lubaszewski, UFRGS
Engenharia de Software: Jaelson de Castro, UFPE
Informática na Educação: Omar Nizram, CTA
Inteligência Artificial: Díbio Borges, UFG
Redes de Comp. e Sist. Distrib.: Wanderley de Souza, UFSCAR
Redes Neurais: Weber Martins, UFG
Sist. Multimídia e Hipermídia: José Valdeni de Lima, UFRGS
Sistemas Tolerantes a Falhas: Ingrid Jansch Pôrto, UFRGS

Sócios Institucionais

Centro de Estudos do Comércio Exterior - Faculdades Positivo
Centro Federal de Educação Tecnológica de Minas Gerais
CEPED - Centro de Pesquisa e Desenvolvimento
CTI - Centro Tecnológico para Informática
FEEVALE - Fed. de Estab. de Ensino Sup. em Novo Hamburgo
FURB - Fundação Universidade Regional de Blumenau
FURG - Fundação Universidade de Rio Grande
IME - Instituto Militar de Engenharia
INPE - Instituto Nacional de Pesquisas Espaciais
ITA - Instituto Tecnológico da Aeronáutica
LNCC - Laboratório Nacional de Computação Científica
PUCCAMP - Pontifícia Universidade Católica de Campinas
PUC-Minas - Pontifícia Universidade Católica de Minas Gerais
PUC-Rio - Pontifícia Universidade Católica do Rio de Janeiro
PUC-RS - Pontifícia Universidade Católica do Rio Grande do Sul
Sociedade Riopretense de Ensino e Educação Limitada
UCS - Universidade de Caxias do Sul
UEM - Universidade Estadual de Maringá
UFAL - Universidade Federal de Alagoas
UFES - Universidade Federal do Espírito Santo
UFF - Universidade Federal Fluminense
UFMA - Universidade Federal do Maranhão
UFMG - Universidade Federal de Minas Gerais
UFPEL - Universidade Federal de Pelotas
UFPR - Universidade Federal do Paraná
UFRGS - Universidade Federal do Rio Grande do Sul
UFRRJ - Universidade Federal Rural do Rio de Janeiro
UFSCAR - Universidade Federal de São Carlos
UFSE - Universidade Federal de Sergipe
UFU - Universidade Federal de Uberlândia
UFV - Universidade Federal de Viçosa
ULBRA - Universidade Luterana do Brasil
UNESP - Universidade Estadual Paulista Júlio de Mesquita Filho
UNICAP - Universidade Católica de Pernambuco

Sumário

CNSBC'98 – SBC&M

invited talks

composições convidadas

artigos

composições selecionadas

tutorial

talks

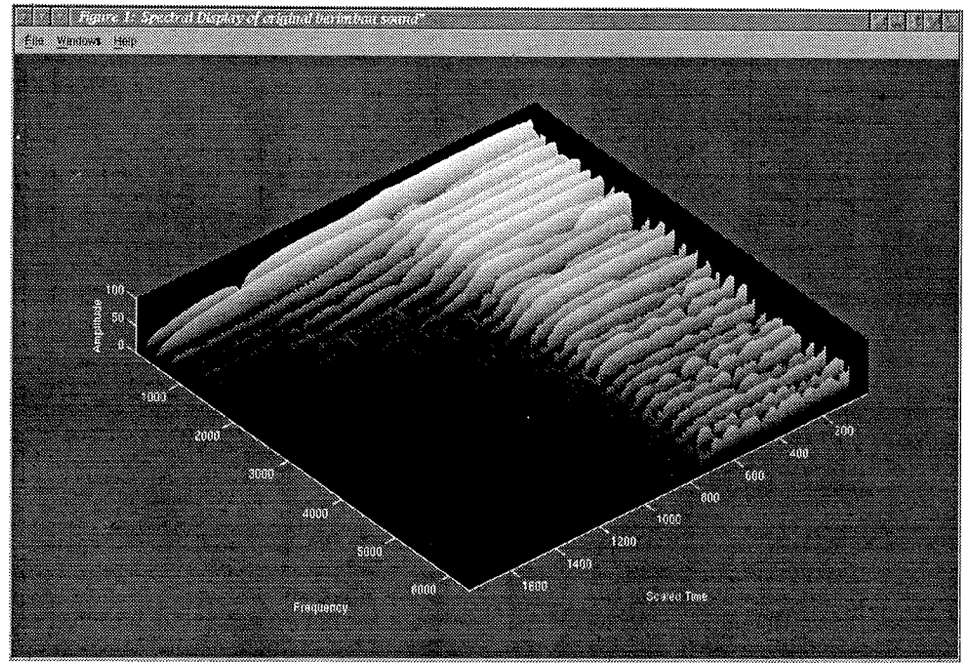
Advances in Analysis-Synthesis for Composition.

Celso Aguiar
CCRMA, Stanford University

The problem of sources separation stands unresolved until today in the field of digital signal processing. Finding its solution can only be measured as or compared to finding a 'holy grail', which could eventually lead to the solution of a series of other related problems in the field.

While the hunt may continue for a long time, and a forthright or even partial answer to the problem remains elusive, the fields of music and composition have both a lot to gain along the quest. The search for its solution has unveiled and continues to unveil many new techniques for synthesis-analysis of sounds and every addition to the repertoire allows for a new insight in the nature and dimension of the problem, while granting control deeper into the sound matter.

In this talk we will take a look at some of these techniques, old and new, and investigate some their direct applications to sound synthesis and transformation in composition.



Projeção Acústica Ortogonal

Aluizio Arcela
Laboratório de Processamento Espectral
Departamento de Ciência da Computação
Universidade de Brasília

Arquiteturas voltadas para performance musical que viabilizam a existência fisica de múltiplos canais de áudio são hoje em dia tecnicamente possíveis graças ao processamento distribuído. Desde que seja possível o posicionamento de uma fonte acústica virtual em qualquer ponto do espaço, o domínio pleno dessa técnica de distribuição de canais de áudio permitirá que a música explore o espaço no seu sentido mais geral.

Nesse contexto, descreve-se uma técnica de projeção acústica que consiste na ação simultânea de duas fontes virtuais — situadas em planos ortogonais: um superior e outro frontal — sobre um mesmo ponto do espaço de audição. Executam-se, neste espaço, notas ortoestereofônicas, as quais são pares de notas que se iniciam no mesmo instante, possuem a mesma duração, e se encerram ao mesmo tempo, sendo que uma delas é executada por um instrumento do plano superior, ao passo que a outra é executada por um instrumento frontal. Instrumento *superior* é um instrumento aditivo contendo n unidades espectrais que soam exclusivamente no plano superior, enquanto que instrumento *frontal* é um instrumento aditivo também contendo n unidades espectrais, mas que soam exclusivamente no plano frontal.

Qualquer que venha a ser a tecnologia utilizada para implementar a ortoestereofonia, é preciso, antes de qualquer providência, que a carta espectral contenha instrumentos verdadeiramente ortogonais, isto é, cada uma de suas unidades-H.deve trazer, além das informações básicas relativas à componente espectral, as coordenadas — (x, z) se for uma nota do plano superior, ou (y, z) se for do plano frontal — da fonte virtual que a emitirá como som.

O recursos tecnológicos disponíveis permitem um *approach* com base em redes e em síntese aditiva via MIDI. Precisa-se de rede porque uma máquina MIDI comum possui apenas duas saídas para o sinal de áudio, e como são oito as linhas necessárias, deve-se fazer uso de quatro computadores-escravos conectados a um computador-mestre que lê a carta espectral; estando cada escravo acoplado a uma máquina MIDI. Precisa-se de MIDI pelas exigências de processamento a tempo real, e precisa-se de síntese aditiva pelo conceito de nota ortoestereofônica.

Uma vantagem adicional de uma rede a quatro máquinas MIDI é que o número total de componentes espectrais que se podem utilizar simultaneamente fica multiplicado por quatro, ampliando-se consideravelmente o espaço de síntese. Uma implementação "alto nível" desta técnica, no que se refere à comunicação entre os computadores, pode ser encontrada com linguagens que possibilitam o manuseio de conexões do tipo cliente-servidor através de *sockets* TCP/IP. Desta forma, pode-se dispor de uma máquina servidora conectada a quatro clientes MIDI em ambiente Internet. Trata-se de um método factível, mas com forte dependência

externa. A solução com base em rede local, por sua vez, torna o processo independente, mas poderá apresentar um sério problema no momento da escolha de uma linguagem que tenha abrangência suficiente para lidar simultaneamente com programação de rede e com os elementos de performance musical.

Alguns resultados teóricos apontam para salas que funcionem como câmeras anecóicas, isto é, totalmente isentas de reverberação, e que sejam capazes de dispor com exatidão um número suficiente de fontes acústicas virtuais em quaisquer coordenadas (x, y, z) do seu interior, e onde não exista nenhuma fonte acústica secundária que não esteja sob total controle do programa de performance.

Understanding and Analysing Electroacoustic Music

Denis Smalley
Centre for Electroacoustic Music Studies
City University, London

As yet we lack a comprehensive, systematic analytical methodology for explaining the rich sonic contexts of electroacoustic music. Our first problem is that the musical qualities most significant to the listening experience cannot suitably be represented in any reliable visual way which might make our task easier. Rather than relying on some widely known, objective system of representation, we have to decide for ourselves what are the pertinent features of the music. This means that in order to diagnose the pertinent features and events which might form the basis of an analytical method, we need to have an understanding of listening behaviours. These pertinences cannot simply be diagnosed by applying rules or by elaborating simple parameters. They are spectromorphological phenomena whose *qualities* have to be teased out aurally, qualities which not only relate to the sounding context of the work, but also refer and allude to sonic and non-sonic experience outside the work: intrinsic and extrinsic features are in intimate liaison. An important goal of analytical exploration is to attempt to reconcile the internal world of the work with the outside world of sonic and non-sonic experience. This is particularly important in acousmatic music, where, in the invisible, spatial play of 'sound-images', there is often an ambiguous entwining of allusion to the real world, and an imaginative distancing from its realities.

In searching for apt words to articulate these pertinences we discover that we inevitably resort to metaphor. Metaphor is a means of travelling between different mental domains, and it provides a fruitful and malleable means of mapping relations between the aural sense, its companion senses, and other forms of human experience.

I find it surprising that we have not made more significant progress in developing the conceptual means for describing and evaluating relations among sounds as they arise in musical contexts. To do that we need to create relational frameworks, which must be archetypal in the sense that they are inherent in music structures, and can therefore be applied across a wide and varied repertory. Furthermore, it is advantageous if these frameworks, once explained, are capable of being grasped by listeners who are not electroacoustic music specialists. In other words, the frameworks should be immediately audible in the music. The use of metaphorical language should aid this communication process at the same time as highlighting the complex nature of intrinsic-extrinsic mapping.

This talk focuses on outlining two relational frameworks. Our musical age is no longer one where we can think of musical 'forms' as such, but we can provide an understanding of the networks of relational frameworks out of which *perceived* form emerges. The first framework

is concerned with the temporal shaping of sonic spectra (spectromorphologies) and how these provide models for perceptions of temporal unfolding. The second framework is concerned with describing the relations among the sonic 'participants' in a work expressed through the metaphor of behaviour. This dual approach is aided by the concept of motion and growth processes, within which are nested the notions of spectral and virtual space. Musical examples will include extracts from works to be heard in the concerts during the Symposium.

Parallel Computing for Musicians

Eduardo Reck Miranda
Department of Music
University of Glasgow, UK
http://website.lineone.net/~edandalex

Abstract

This article introduces the fundamentals of parallel computing and discusses its benefits for musicians. It begins by introducing classic parallel architectures and parallel programming strategies. Next, it discusses the benefits of parallel computing for a specific area of computer music research: sound synthesis. The article then concludes with a brief discussion on the future of the musician's desktop computer. It is suggested that new paradigms for computer music will emerge from parallel computing technology, but it is up to our community to find the right approaches to them.

1. Introduction

Since computers were invented there have been no major changes in their basic design. The main core of most currently available computers is a central processing unit (CPU), provided with a single memory component, which executes instructions sequentially. This configuration is commonly referred to as *von Neumann architecture* (after its inventor), but for the purposes of this article I use the more generic acronym SISD, for Single Instruction Single Data.

SISD machines call for the notion of sequential programming in which a program consists of a list of instructions to be executed one after the other. Although such machines may now work so fast that they appear to be executing more than one instruction at a time, the sequential programming paradigm still remains for most techniques and tools for software development.

The sequential paradigm works well for most of the ordinary tasks we need the computer for, but scientists are very aware of the limitations of current computer technology. Assuming that computers were programmed to display some sort of intelligent or customary behaviour, the sequential paradigm would fail to model many aspects of human intelligence and natural systems. For example, particular mental processes seem to be better modelled as a system distributed over thousands of processing units, as an idealised model of brain tissue. It is possible to simulate this distributed type of behaviour on fast SISD machines by modelling a limited number of processing units, but no single processor nowadays can support the number of units required to study the behaviour of the more sophisticated distributed machine models.

Although the speed of SISD machines has increased substantially over the last few years, these improvements are beginning to touch the limits of physics, beyond which any further

development would be impossible. In particular, the speed of light is a constant upper boundary for the speed of signals within a computer, and this speed is already proving too slow to allow further increases in the speed of processors. A plausible alternative to meet the demand for increasing performance is to abandon dependence upon single processor hardware and look towards parallel processing techniques.

Parallel computing is commonly associated with the possibility of breaking the speed limits imposed by SISD machines. However much faster computers become, parallel computing also offers interesting programming paradigms for the development of innovative computer music software; see, for example, Holm's paper on the applications of the Communicating Sequential Processes (CSP) technique in music composition and analysis (Holm, 1992). From a musician's perspective, it is this latter aspect of parallel computing that excites interest here.

2. Architectures for parallel computing

Two main approaches are used to build parallel computers: SIMD (for Single Instruction Multiple Data) and MIMD (for Multiple Instructions Multiple Data). Whilst the former employs many processors simultaneously to execute the same program on different data, the latter employs several processors to execute different instructions on different data. Both approaches underpin different programming paradigms and have their own merits and inadequacies.

2.1. The SIMD architecture

SIMD-based computers employ a large amount of interconnected *processing elements* (PE) running the same programming instructions concurrently, but working with different data (Figure 1). Each PE has its own local processing memory, but the nature of the PEs and the mode of communication between them varies for different implementations. PEs are usually simpler than conventional processing units used on SISD machines because PEs do not usually require individual instruction fetch mechanisms; this is managed by a master control unit (MCU). Communication between PEs commonly involves an orthogonal grid of data pathways and each PE can often communicate directly with its four or eight nearest neighbours.

SIMD computers are essentially synchronous models because all PEs execute the same instruction at the same time. The only control the programmer has over an individual PE is to allow or prevent it from executing the instruction. This makes SIMD computers easier to program than MIMD computers (see below) because the control of different programs running on each processor of a MIMD machine becomes very demanding as the number of processors increase. SIMD architectures therefore suit the provision of very large arrays of PEs; current SIMD computers may have as many as 65,536 PEs.

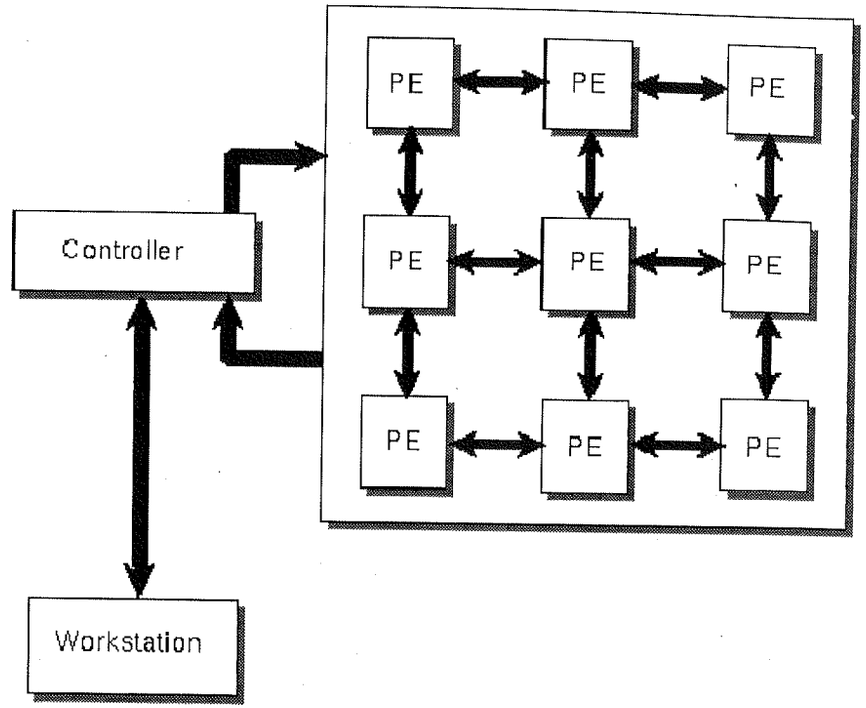


Figure 1
The SIMD architecture employs a large amount of interconnected rocessing elements running the same instructions concurrently.

2.2. The MIMD architecture

MIMD-based computers employ independent processors running different programming instructions concurrently and working on different data. There are a number of different ways to construct MIMD machines, according to the relationship of processors to memory and to the topology of the processors.

Regarding the relationship between processor and memory, we identify two fundamental types: *shared global memory* and *private local memory*. In shared global memory, each processor has a direct link to a single global memory via a common pathway, or *bus* in computer science jargon (Figure 2). In this case, processors only communicate to each other via the global memory. This type of MIMD architecture is sometimes preferable because they are relatively straightforward to program. The major drawback is that the number of processors should be limited to accommodate the capacity of the bus. Otherwise, too many processors competing to access the memory would need onerous mechanisms to prevent data traffic jams.

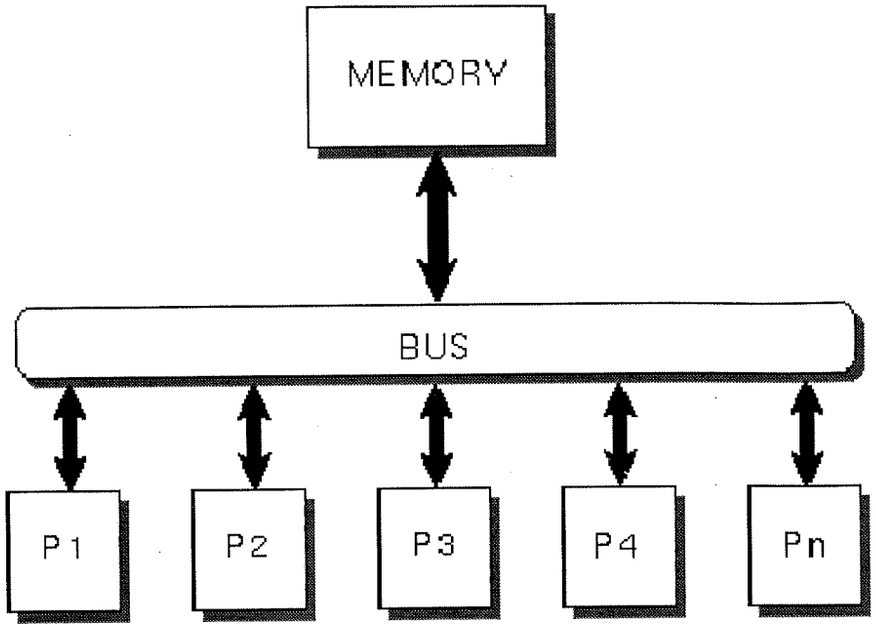


Figure 2:
In MIMD's shared global memory each processor has a direct link to a single global memory.

In order to employ large amounts of MIMD processors it is necessary to furnish each processor with its own local memory. In this case, each unit is normally encapsulated in a microchip called a *transputer.*

Since MIMD transputers execute different instructions and work on different data, the interconnection between them must be carefully planned. It is utterly unthinkable to connect each transputer to all other transputers. The number of connections rises as the square of the number of units. From a number of possibilities, most customarily used topologies are *binary trees, two dimensional grids* and *hypercubes* (3). Sophisticated MIMD implementations allow for user-configuration of topologies to suit particular needs.

MIMD computers are essentially asynchronous because each processor or transputer may proceed at their own rate, independently of the behaviour of the others. They may, of course, perform in synchronous mode if required.

3. Parallel programming strategies

It seems likely that in a few years time computers will be based upon a parallel architecture of some kind. Some of these machines will probably make their parallel architecture transparent to high-level programmers and will somehow allow for the processing of sequential programs as well. In this case, sequential programs must to be adapted either manually by a programmer or automatically by the machine itself (e.g. by the operational system).

Cutting edge technology and best achievable performance will certainly require software explicitly designed for parallelism. There are still many intrinsically sequential problems that pose difficulties for the design of a parallel processing solution. For example, a melody implies sequences of sounds, which in turn imply sequences of samples; these are best processed sequentially. A program that has been specifically designed for a parallel architecture, will perform unquestionably better than a converted sequential program. This is to say that there certainly are some problems that suit parallelisation better than others - it is up to us software developers to make the right choices.

There are two different but closely related parallel programming strategies: *decomposition* and *parallelisation.* The former involves methods to decompose a sequential program for parallel processing and the latter involves methods to design parallel programs from scratch.

All these strategies are inter-related and very often the solution to a problem is best achieved by applying a combination of them.

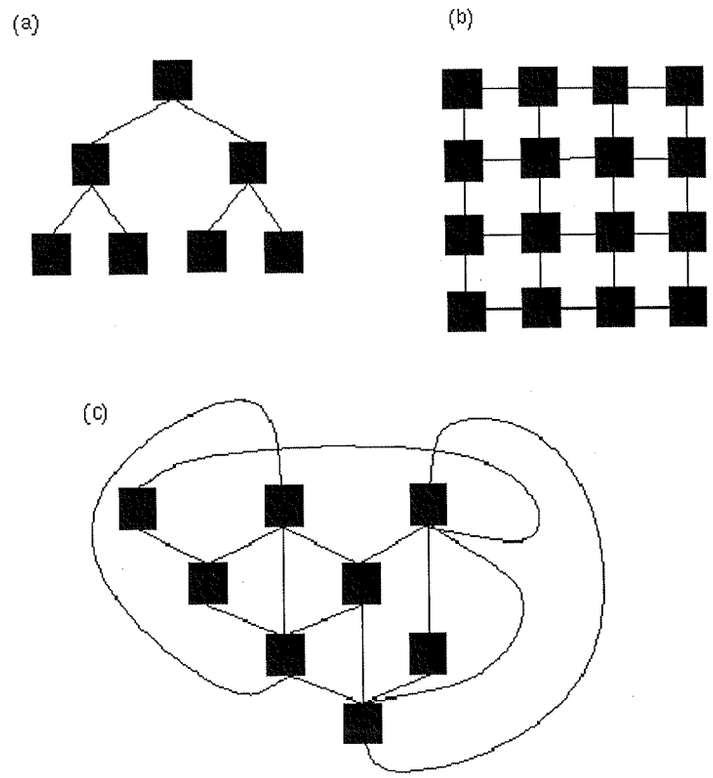


Figure 3
Most customarily used MIMD topologies: (a) binary trees, (b) two dimensional grids and (c) hypercubes.

3.1. Decomposition

The main objective of decomposition is to reduce the execution time of a sequential program. In general, it is possible to split a sequential program into various parts, some of which would have good potential for concurrent processing. By allocating these parts to various parallel processors, the runtime of the overall program could be drastically reduced.

Decomposition requires caution because the splitting of a program into a large number of processors incurs considerable costs. For example, it should be estimated that individually processed parts might require demanding communication mechanisms to exchange data between them. That is, a large amount of processors does not necessarily lead to good performance.

I identify three main strategies for decomposition: *trivial*, *pipeline* and *geometric*. A trivial decomposition takes place when the original sequential program processes a great amount of data with no interdependencies. In this case, there are no major technical problems to split the task, since parallelism may be achieved simply by running copies of the entire program on various processors concurrently. Conversely, a pipeline decomposition strategy is applied when a program can be split into various modules and allocated to different processors. All input data still passes through each of the modules sequentially but as they move through the pipeline, the modules work simultaneously (Figure 4). The first input element passes through the first stage of the pipeline and after being processed, it then passes on to the next stage of the pipeline. While this first element is being processed at the second stage, the second input element is fed into the first stage for processing, and so on. All stages work simultaneously because each has its own exclusive processor.

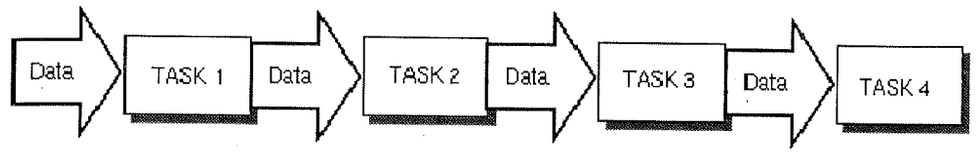


Figure 4:
In pipeline decomposition the input data passes through each of the modules but as they move through the line the modules work simultaneously.

Finally, geometric decomposition splits the data rather than the program. This strategy is suited for those problems where identical operations are applied to different parts of a large data set. As with the case of trivial decomposition, each processor has a copy of the whole program. The difference between trivial and geometric decomposition is that the latter is designed to cope with data dependencies; that is, the result of the operation of one data subset may require information about the operation of other data subsets. Geometric decomposition works best when the data subsets need to interact with their neighbouring subsets. In this case the data is distributed to a certain number of processors arranged in a lattice and the program will have to be furnished with mechanisms for passing messages across the processors.

3.2. Parallelisation

There are two main different approaches to the design of a parallel program: *specialist processing* and *task farm processing*. These approaches have many links to the decomposition categories introduced above and both concepts certainly complement each other.

3.2.1. Specialist processing

Specialist processing achieves parallelism by dividing the problem into specific types of tasks. For example, if the problem is to build a piano, then there will be a variety of specialised tasks such as making the keys, making the string, making the case, etc. Once all pieces are complete,

the piano is then assembled. Specialist workers are assigned for each task in order to assure the quality of the components and foster the speed of construction. However, if the goal is to build only one piano, workers may not be able to start working simultaneously because some will have to wait for the outcome of other colleagues in order to start their job. Conversely, if the goal is to manufacture several pianos, then it may be possible to keep all workers occupied for most of the time by 'pipelining' the tasks (see pipeline decomposition above). In any case, it is essential to establish efficient channels of communication between the workers so that they can co-ordinate their work.

3.2.2. Task farm processing

In task farm processing, the problem is divided into several tasks but not necessarily targeted for specialist processors. To continue the aforementioned metaphor of the piano, there are no specialist workers in the team here, because it is assumed that all of them are capable of carrying out any of the tasks for building a piano. In this case, there is a master who holds a list of tasks to be performed. Each worker takes a task and when it is accomplished he or she comes back and picks another one. Communication occurs mostly between a worker and the master; for the sake of maximum performance, workers are not encouraged to talk to each other because their attention must not be diverted from main task.

4. Benefits for music: sound synthesis

The benefits of parallel computation for music can be enormous but here I will illustrate only its application in sound synthesis systems. In several ways, parallel computing concepts already have started making their way into synthesis technology, from the design of dedicated VLSI (Very Large Scale Integration) chips to the implementation of new paradigms for sound synthesis.

4.1. Signal processing level

Dedicated digital signal processors (called as DSP) with multiple functional units and pipeline methods have been encapsulated into VLSI chips and today are commonly found in a variety of hardware for sound synthesis; e.g. the Betel Orionis system developed at the University of Rome in Italy [Nottoli and Costantini, 1998].

The functioning of a DSP is driven by a set of instructions that is loaded into its own memory from a host computer. The DSP then cycles continuously through these instructions until it receives a halting message; a sample is produced at each cycle of the DSP. Special DSP arrangements may form arrays of DSP for processing blocks of samples at once. In this case, the output for each cycle is an array of samples.

Also, parallel configurations of specially designed general purpose microprocessors based on RISC technology (Reduced Instruction Set Computer) have been used on several occasions; e.g. the IRCAM Musical Workstation board developed in France [Lindermann et al., 1991].

4.2. Software synthesis programming level

On the software synthesis programming front, there have been a few attempts to decompose existing synthesis programming languages for concurrent processing.

Remarkable results have been reported by composer Peter Manning and his collaborators at Durham University, in England, who have managed to run Csound concurrently on a parallel machine [Bailey et al., 1990]. Csound is a synthesis programming language in which instruments are designed by connecting many different synthesis units. One or more instruments are saved in a file called the *orchestra*. Then the parameters to control the instruments of the orchestra are specified on an associated file called the *score* [Miranda, 1998]. This score file is organised in such a way that each line corresponds to a stream of parameters to produce a sound event (or note) on one of the instruments of the orchestra. For instance:

```
i1  0  1.0  440  90
i2  1  0.5  220  80
...
```

In the example above, each line specifies five parameters for two different instruments, including the 'names' of the instruments (i.e., *i*1 and *i*2, respectively), start time (in seconds), duration of the note (in seconds), frequency (in Hz) and amplitude (in dB). Each line produces a note.

The parallel implementation of the Durham team is inspired by the *task farm* paradigm discussed earlier. A copy of the entire orchestra is distributed to each processor and the score is considered as a list of tasks for the processors; that is, each processor picks a line from the score for processing.

4.3. Synthesis modelling level

At this level a parallel computing paradigm is normally imbedded in the synthesis model itself. As an example I cite *Chaosynth*, a synthesis system developed by myself in collaboration with the engineers of the Edinburgh Parallel Computing Centre (EPCC), in Scotland [Miranda, 1994] [Miranda, 1995].

In short, *Chaosynth* uses cellular automata (CA) to control the parameters of a granular synthesis instrument. The granular synthesis of sounds involves the production of sequences of thousands of short sonic particles (for example, 35 milliseconds) in order to form larger sound events.

CA are mathematical models of dynamic systems in which space and time are discrete and quantities take on a finite set of discrete values. CA are often represented as a regular array with a discrete variable at each site, referred to as a cell. The state of the CA is specified by the values of the variables at each cell. It evolves in synchronisation with the tick of an imaginary clock, according to a global function that determines the value of a cell based upon the value

of its neighbourhood. As implemented on a computer, the cells are represented as a grid of tiny rectangles, whose states are indicated by different colours (Figure 5).

The algorithm of the CA used in *Chaosynth* is fully explained in [Miranda 1995]. Each sound granule produced by *Chaosynth* contains a number of partials produced by a bank of oscillators, as with additive synthesis. The frequencies of the partials are determined according to the evolution of the CA. The states of the CA cells represent frequency values rather than colours. The oscillators are associated to a group of cells, and in this case the frequency for each oscillator is established by the arithmetic mean of the frequencies of the cells associated to the respective oscillator. At each cycle of the CA, *Chaosynth* produces one sound granule and as the CA evolves, the components of each granule change therein. The size of the grid, the number of oscillators and other parameters for the CA are all configurable by the user.

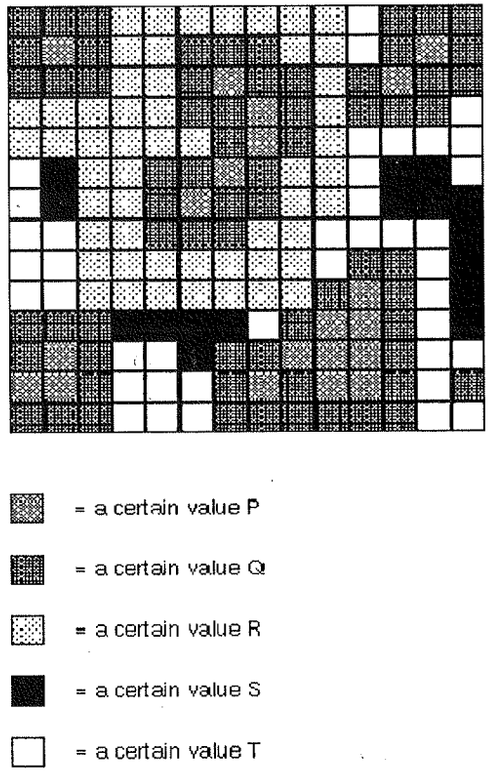


Figure 5:
The CA cells are represented as a grid of tiny rectangles, whose states are indicated by different colours.

CA are intrinsically suited for parallel processing; they are arrangements of cells, each of which is updated every cycle, according to a specific global rule that takes into account the state of the neighbouring cells. This is a typical problem for geometrical decomposition with message passing. In this case, the grid is subdivided into a number of contiguous rectangular portions in such a way that all of them are handled concurrently by different processors. The oscillators of Chaosynth are therefore computed in parallel (Figure 6).

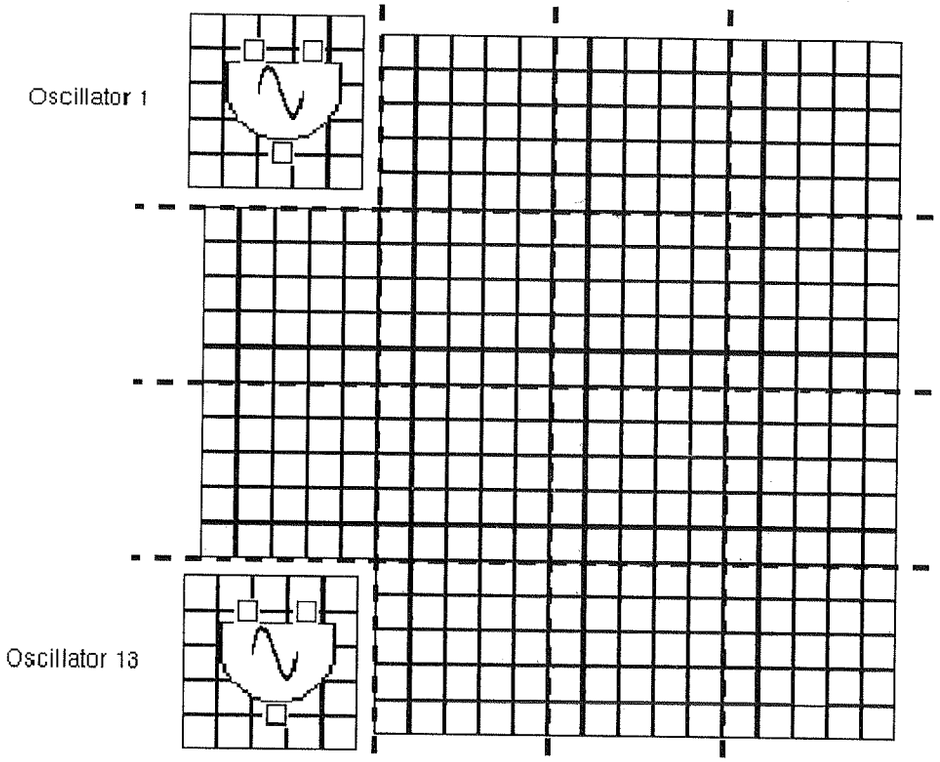


Figure 6:
CA are suited for geometrical decomposition.

5. Conclusion

Since the invention of the microprocessor, computers have become twice as fast almost every two years; the microprocessor of your average personal computer nowadays encapsulates more than 7 millions of transistors in a single silicon chip. Because of this explosive progress, current computers are millions of times more powerful than their crude ancestors, but integrated circuit

technology is running up against its limits. Advanced lithographic techniques would certainly be able handle circuitry designs 1/100 the size of what is currently available, but at this scale - at the atom level - integrated circuits cannot function properly. If computers are to become more powerful and yet smaller, new technology must either replace or supplement current trends.

Research in quantum-mechanical computer technology is very promising but until now scientists did not manage to actually build a suitable quantum processor for industrialisation. In meantime, the industry is supplementing the power of current microchip technology with parallel computing strategies; e.g. pipelining is becoming increasingly popular in microchip design.

In the future, musicians may never want to know whether his or her computer employs parallel processing concepts. All the same, parallel software engineering techniques are becoming more popular and new parallel paradigms for computer music will certainly continue to emerge. It is up to our community to develop them.

References

[Bailey et al. 1990] Bailey, N., Purvis, A., Bowler, I. and Manning, P., "Concurrent Csound: Parallel Execution for High Speed Direct Synthesis", Proceedings of the International Computer Music Conference, Glasgow: ICMA, UK.

[Holm, 1992] Holm, F., "Machine tongues XIV: CSP-Communicating Sequential Processes", Computer Music Journal, 16(1):25-33, MIT Press, USA.

[Lindermann et al. 1991] Lindermann, E., Dechelle, F., Smith, B. and Starkier, M., "The architecture of the IRCAM musical workstation", Computer Music Journal, 15(3):41-49, MIT Press, USA.

[Miranda, 1994] Miranda, E. R., "Chaosynth - computer music meets high-performance computing", Supercomputer, 11(1):16-23, ASFRA, The Netherlands.

[Miranda, 1995] Miranda, E. R, "Chaosynth: Um sistema que utiliza um autômato cellular para sintetizar partículas sônicas", Anais do II Simpósio Brasileiro de Computação e Música, Porto Alegre: Instituto de Informática da UFRGS, pp. 205-212, Brasil.

[Miranda, 1998] Miranda, E. R., Computer Sound Synthesis for the Electronic Musician, Oxford (UK): Focal Press.

[Nottoli and Costantini, 1998], Nottoli, G. and Costantini, G., "Betel Orionis: a real-time, multiprocessing sound synthesis system", Actes des Journées d'Informatique Musicale 98, Marseille: Publications du LMA no. 148, pp. E3-1-E3-5, France.

The Theory and Practice of Micro-Sound

Curtis Roads
CREATE
Department of Music
University of California Santa Barbara

The evolution of musical expression is intertwined with the development of musical instruments. This has never been more evident than in the 20th century. Beginning with the gigantic Telharmonium synthesizer unveiled in 1906, research has ushered forth a steady stream of electronic instruments that have irrevocably molded the musical landscape.

The most precise and flexible electronic music instrument ever conceived is the digital computer. Like the pipe organ invented centuries earlier, the computer's power derives from its ability to emulate, or in scientific terms, to model phenomena. Unlike the pipe organ, the computer's models are expressed in symbolic code. Thus it does not matter whether the phenomena being modeled exist outside the circuitry of the machine, or whether they are pure fantasy. This makes the computer an ideal testbed for the representation of musical architecture.

Music theory has long recognized a hierarchy of structure in music compositions. A central task of music composition has always been the management of the interaction amongst structures on different time scales. Starting from the topmost layer or form and proceeding downward, one can dissect layers of structure, arriving at the bottom layer of individual notes.

This hierarchy, however, is incomplete. Beneath the level of the note lies another multi-layered stratum, the micro-sonic hierarchy. Like the quantum world of the quarks, leptons, gluons, and bosons, the micro-sonic hierarchy remained invisible for centuries. Modern tools let us view and manipulate the micro-sonic layers.

Here we distinguish six time scales, starting from the longest, corresponding to different levels of perception.

1. Macro - The overall musical architecture or form, measured in minutes or hours.

2. Meso - Groupings of objects into hierarchies of phrase structures of various sizes, measured in minutes or seconds.

3. Sound object - A basic unit of musical structure, generalizing the traditional concept of note to include complex and mutating sound events on a time scale ranging from a fraction of a second to several seconds.

4. Micro - Sound events on a time scale that begins at the object level and extends down to the thresholds of auditory perception (measured in thousandths of a second or milliseconds).

5. Sample - The "atomic" level of digital audio systems: individual binary samples or numerical amplitude values, one following another at a fixed time interval. The period between samples is measured in millionths of a second (microseconds).

6. Sub-sample - Fluctuations on an infinitesimal time scale, too brief to be properly recorded or perceived, measured in billionths of a second (nanoseconds) and ranging down to the infinitesimal durations.

This presentation examines this temporal hierarchy, paying particular attention to the micro-time scale. We explore new methods of sound synthesis and processing that operate on the micro-time level, and study the aesthetic implications of these techniques. The presentation will be accompanied by diagrams and sound examples.

composições convidadas

CÍRCULOS CEIFADOS (1996-97)

(Em fita magnética e texto com ilustrações,
para leitura ou conferência)

Rodolfo Caesar
Escola de Música
Universidade Federal do Rio de Janeiro
rc@unisys.com.br

O conjunto apresentado combina duas partes complementares: uma peça eletroacústica e um livro.

O livro pode ser usado como material para conferência introdutória à música - eletroacústica e acusmática, principalmente - com o propósito de atrair a atenção de ouvintes para o trabalho do compositor em seu ato de composição. Acreditando que, ao invés de arrefecerem a complexidade de seus trabalhos, os compositores deveriam comunicá-la aos ouvintes na medida do possível, assim tentei recorrendo a ilustrações gráficas e literárias. Aceitas as limitações para não ultrapassar a conveniência de um projeto pedagógico, cada passo importante e a estratégia composicional são, no texto, descritos por meio de vocabulário aproximativo, metáforas e ilustrações, com a fantasia como fator de engajamento na leitura e como crítica ao 'peṣquisadorismo' institucional.

Baseado em minhas pesquisas sobre sons e ritmos da bio-acústica através de Modulação de Freqüência e do programa Csound, este projeto desenvolve argumento improvável jogando com o falso e o verdadeiro. O texto afirma que os fenômenos conhecidos como Crop Circles (círculos misteriosamente ceifados em campos de cereais) poderiam ser compreendidos à luz de sua semelhança com as formas espiraladas dos abalones, moluscos em forma de orelha, segundo modelos desenhados pelo sábio Mendes da Costa, quando este viveu na prisão na Inglaterra no século dezessete. As explicações recorrem à série fibonaciana e a análises espectrais, passando por comentários sobre o mistério acusmático na experiência da escuta eletroacústica. A música funciona como chave e comprovação dessa teoria.

A peça foi composta em meu estúdio particular, usando programas como Csound e SoundHack, e outros programas gráficos para visualização de conteúdos espectrais.

Encomenda do programa de Bolsas da Fundação Vitae 1995.

Galileo's First Glimpse (1996)

(video version of an interactive work)

Craig R. Harris
The Leonard Almanac, Minneapolis, USA
harri067@tc.umn.edu

Galileo's First Glimpse (1996) characterizes the moment when Galileo first viewed through his first telescope. In an instant the world that has been intersects with the world of possibilities. The synapse created in their intersection transforms perception, and the world can never be seen in the same way again. Graphics are created by Lorren Stafford.

Galileo's First Glimpse is a live, interactive performance work. The music is performed using a digital audio workstation and a digital sampler. The stereo sound tracks serve as the source input for an interactive graphics generation system, which is projected onto large screens during performance. In addition to the live music performer, a live graphics performer controls visual transformations of the audio signal, affecting the color pallete, waveshapes, and motion. The video version of *Galileo's First Glimpse* had its premiere at the 1996 International Symposium on Electronic Art, and has since been presented at the Art on the Electronic Edge festival, at the 1997 SEAMUS national conference, at several locations throughout the United States and in Brazil.

Wind Chimes (1987)

(for tape)

Denis Smalley
Centre for Electroacoustic Music Studies
Music Department
City University, London
d.smalley@city.ac.uk

The main sound source for 'Wind Chimes' is a set of ceramic chimes found in a pottery during a visit to New Zealand in 1984. It was not so much the ringing pitches which were attractive but rather the bright, gritty, rich, almost metallic qualities of a single struck pipe or a pair of scraped pipes. These qualities proved a very fruitful basis for many transformations which prised apart and reconstituted their interior spectral design. Not that the listener is supposed to or can always recognise the source, but in this case it is audible in its natural state near the beginning of the piece, and the ceramic quality is never far away throughout. Complementary materials were gathered to expand the piece's sound-families, among them very high metallic Japanese wind chimes, resonant metal bars, interior piano sounds, and some digital synthesis. The piece is centred on strong attacking gestures, types of real and imaginary physical motion (spinning, rotating objects, resonances which sound as if scraped or bowed, for example), contrasted with layered, more spacious, sustained textures whose poignant dips hint at a certain melancholy.

'Wind Chimes' was commissioned by the South Bank Centre, London, and was given its first performance in the Electric Weekend at the Queen Elizabeth Hall. The computer transformations were carried out in the Digital Studio of the Groupe de Recherches Musicales, Paris, and the piece was mixed in the Electroacoustic Music Studio at the University of East Anglia. label.

Clarinet Threads (1985)

(for clarinet and electroacoustic sounds)

Denis Smalley
Centre for Electroacoustic Music Studies
Music Department
City University, London
d.smalley@city.ac.uk

The clarinet can produce a variety of sound-types - key noises, air sounds, degrees of sound production producing less definite pitches, very high notes produced by biting the reed, multiphonics. The electroacoustic medium provides an opportunity for integrating this sound repertory into an expanded sonic environment. Thus the clarinet is threaded through the electroacoustic fabric, sometimes merged with it, sometimes surfacing in a more soloistic role. Besides passages which use the clarinet in a traditional manner there are stylised environments drawn from outside music - the calls and cries of nature, the movement of wind and water, and textural motion suggesting floating and drifting.

Most of the electroacoustic sounds were created during visits to a variety of studios in the early 1980s - the Computer Systems Research Institute at the University of Toronto (the SSSP system), the Finnish Radio Experimental Studio, the Groupe de Recherches Musicales Digital Studio, and the University of Birmingham Electroacoustic Music Studio. Mixing was carried out at the University of East Anglia. 'Clarinet Threads' was commissioned by Roger Heaton with funds provided by Eastern Arts, England, and was first performed in the Norwich and Norfolk Festival in 1985. In 1988 it was awarded the Prix Ars Electronica.

Valley Flow (1992)

(for tape)

Denis Smalley
Centre for Electroacoustic Music Studies
Music Department
City University, London
d.smalley@city.ac.uk

The formal shaping and sounding content of 'Valley Flow' were influenced by the dramatic vistas of the Bow Valley in the Canadian Rockies. The work is founded on a basic flowing gesture. This motion is stretched to create airy, floating and flying contours or broad panoramic sweeps, and contracted to create stronger physical motions, for example the flinging out of textural materials.

Spatial perspectives are important in an environmentally inspired work. The listener, gazing through the stereo window, can adopt changing vantage- points, at one moment looking out to the distant horizon, at another looking down from a height, at another dwarfed by the bulk of land masses, and at yet another swamped by the magnified details of organic activity.

Landscape qualities are pervasive : water, fire and wood; the gritty, granular fracturing of stoney noise-textures; and the wintery, glacial thinness of sustained lines. The force and volatility of nature are reflected in abrupt changes and turbulent textures.

The mixing of the piece was started during a Creative Residency in the Media Arts Program at the Banff Centre for the Arts situated in the Bow Valley. Sounds previously created at IRCAM were incorporated, and further materials were subsequently developed at Simon Fraser University. The piece was completed in the composer's studio. 'Valley Flow' was commissioned by the Birmingham Electroacoustic Sound Theatre (BEAST) with funds provided by West Midlands Arts, and was given its premiere in a live BBC broadcast.

artigos

Modelling musical cognitive processes with the use of supercomputers

Francesco Carreras
CNUCE/CNR via S.Maria 36, 56126 Pisa, Italy
e-mail: F.Carreras@cnuce.cnr.it

Abstract

The development of realistic cognitive models requires computing resources that the progress of technology and the use of parallel processing techniques made available in the last few years in many research institutions. In this paper the results of the modelling of two different musical cognitive environments are described.

Introduction

In recent years the modelling of cognitive processes has become a research goal in many institutions around the world. From one side this interest has been stimulated by the new developments in neurology, psychology, brain and cognitive sciences and by a renewed attention to the phenomena connected to music perception; e.g. see [Kru90]][La92][La97][Sei91]. On the other side a tremendous progress in computer hardware technology made cheap computing power available in many research laboratories. A realistic model for the simulation of perceptive and cognitive processes requires large amounts of computer resources both in terms of memory capacity, computing power and large disk storage space. The availability of supercomputers and parallel computers with usable software development environments made it possible to design, implement and use, also without the assistance of computer specialists, of large simulation models in the musical domain. Various parallelizing tools and primitives, such as for instance Express, MPI, MPL, allow the transformation of programs written in languages like Fortran or C from a batch version into a parallel one with limited effort and skill. Application programs analysis and decomposition techniques are becoming of common use in the scientific community.

Given these developments a shift of attention was possible from the difficulties of implementation to the modelling process in itself . Large models with high memory and computing requirements, which can be decomposed and distributed over many different

processors of the same parallel system, can be designed without particular attention to the computing environment constraints. The latter have indeed become are less and less restrictive during the past decade. A consequence is that the implementation of larger and more realistic models is more feasible than untill a few years ago. The possibility of controlling many different consoles from one single screen allows more productive ways of conducting a project by having several applications running at the same time on different processors of the same computer. In the following paragraphs two different cognitive models that were implemented on a parallel computer will be presented, with special focus on their technological side.

Cognitive models for simulating music perception

A theory of music cognition has recently been developed by M. Leman [Le95] and a computer model was realised with the aim of verifying the basic assumptions of the theory. A central point of the theory is that the tonal relationships imbedded in the western tonal music are learnt by repeated exposition to that type of music so that internal memory representations of those relationships develop by adaptation to the musical environment. The model employs sampled real music which is filtered and processed by an auditory model, that simulates the perceptive transformations of the human auditory system. A self-organizing neural network, SOM [Ko95], is then used to extract high-level implied structural information out of these images in terms of a mapping onto a two-dimensional grid. In particular the SOM is trained with so-called context images, which are obtained by integration over a fixed time window of a sequence of contiguous stable structures called completion images and fed into the network in random order. Through a process of self-organization schemata are developed that represent tonal information in terms of a topology that represents the circle of the fifths. This approach is called "ecological" for the reason that the role of the programmer is limited to the specification of the interactions between the musical environment and a model of perceptual learning [Le96][Le97].
A second model has been designed and implemented [Ca98A,B] that makes use of some of the components used in the previous model. Here again an ecological approach is followed. This means that the data are real sampled music that no score-based or symbolic information is used and that the results develop by self-organization. The aim of this model is to extract of the harmonic information embedded in a musical piece. The classification of the possible chord configurations used for training a neural network follows a recent research proposal that is part of an original harmonic theory by L.Balsach [Ba97]. For each the 12 notes of an octave 11 so-called convergent chords are assigned according to precise criteria stated in the theory [Ba94]. In our modelling approach these chords are sampled and processed by the auditory model previously mentioned. The resulting completion images are then passed through an onset detection module [Mo97] that is able to precisely define the perceived onset of all the musical events of the musical piece. A choice is made of a representative part (40 samples of 4 ms each) of every musical event which, in

this case represents one of the 132 chords. A SOM is then trained and a map is designed that shows the responding regions for each of the training chords (being there represented by 40 points). The process of chord classification is then carried on with a piece of real music that is to be passed through the trained neural network acting as an analyser of the input vectors configurations. The final outcome is a table that contains, for each musical event, the chords that respond with a value above a fixed threshold. Each of the responding chords has a weight that represents the degree of fit with the given event. The analysis of these results can be interpreted as a harmonic analysis of the musical piece.

The use of parallel computers in the simulation of large cognitive models

The first model mentioned above was gradually developed over several years. The first version was designed and implemented by M.Leman on a four T800 transputer system using 3L-C and then Express [Le89]. The hardware limitations of the system allowed only small scale simulations. In 1994 the model was ported on a nCUBE2 parallel system hosted at CNUCE, Pisa. Different configurations were tested and monitored with the aim of finding the best trade-off between computing and communication load. A final configuration of eight 16 Mbyte processors was selected to perform a simulation with a 20 by 20 grid and 2000 vectors of 56 components each, for a total of 360000 vectors because the simulation process was run for several epochs. The execution required almost 24 hours and the balance between communication and execution time was 51,5% and 48,5% respectively. The third set of simulations was then performed on an IBM SP2 parallel system of CNUCE. The model was completely rewritten by F.Carreras and M.Leman [Ca96] in order to exploit the memory capacity and the power of the new system. The new program was parallelized with MPL and was designed so as to be run in different processor and neural grid configurations The goal this time was ambitious. The simulation was to use as input data the sampled complete first book of the Das Wohltemperierte Klavier by J.S.Bach. The images (56 data points each) employed were over 380000 to be randomly used for several epochs. Also the employed grid was now larger to obtain a finer resolution. A 30 by 30 and a 100 by 100 grid were used. The idea at the base of this implementation was to assign each processor part of the grid. This would then communicate the value of the most responding neuron, along with its coordinates, to all other nodes. The absolute most responding neuron was then chosen and each node operated the due adaptation for the neurons that were into its assigned part of the grid, having in mind that the grid was conceived as a torus shape, i.e. each edge is connetcted to its opposite edge. Several simulations were performed with the objective of finding the set of system parameters that would produce the highest speed-up. The SP system can work in IP (Internet protocol mode) whereby the communication between nodes uses an Internet like protocol. This mode is slower than the US (User Space) mode but allows several users to share the same node. The latter protocol makes use of a High Performance Switch that considerably increases the

internode communicatrion speed, but requires dedicated nodes. A table reports the data of different parameters configurations for the same simulation.

Table 1

PROCS	CMODE	TELAP	TCOMM%	SPEEDUP	
1	-	633.92	0	1	
2	IP	440.28	24.65	1.43	
2	US	363.76	9.78	1.75	
4	IP	381,07	53.56	1.66	
4	US	205.49	18.17	3.08	
8	IP	172.71	47.08	3.67	
8	US	118.12	16.07	5.37	

Where PROCS is the number of processors; CMODE is the User Space or Internet Protocol; TELAP is the clock elapsed time (with dedicated system); TCOMM% is the percentage of interprocessor communication time and SPEEDUP is a measure of the degree of parallel processing.

The configuration that assured a balance between speed-up and shared use of the system was chosen and most of the simulations were performed with 4 processors in US mode. Each epoch, with 380.000 vectors, required about 5 hours of elapsed time. The results of the simulations, of great interest from a theoretical point of view, were reported in several presentations [Le95][Le96][Ca96].

The use of supercomputers for speeding up musical modelling research

In the second project the supercomputer was used in a different way. Now the volume of the data involved in the simulations is low and the issue is not in system speed but rather in the necessity of performing many different tasks in a short time and if possible at the same time. A chain of programs, that represents the perceptive and cognitive processes, must be repeatedly perfomed. The chain is made of an auditory model, a periodicity analysis autocorrelation step, an on-set analysis, a neural network training or testing. A critical problem was the determination of the type of musical data to be used for training the network. Synthetic Shepard tones [She64] were used first, and real music data, recorded with different techniques, were then tried. Each simulation runs for about seven hours on one SP2 node. A second chain of computations was tied to the creation of 495 four-note chord configurations to be then tested with the trained network, whose results for each chord were subsequently processed to produce a table with chord decompositions. The

work plan encopassed the preparation of many different sound configurations, in order to compare the chord decomposition results, and choose the highest scored one for the analysis of musical examples. The two chains of programs had to be performed many times, and data acquisition and preparation, programs adjustements and data analysis were some of the many tasks required.

The solution that was adopted to increase the productivity of the research process was to choose a remote workstation equipped with an X Window System simulator. In the specific case eXodusPowerPC™ was utilized. Short-cut connections to different nodes of the supercomputer were defined. The workstation allowed for the simultaneous use of tens of remote consoles and edit screens. With a setup of this kind one console was used to start and follow a simulation process; an edit screen could then show partial results or visualization of map layouts [Kra92]; at the same time development work was going on from another console that could utilize other edit screens. Data acquisition from an external musical keyboard was performed directly from the workstation as well as resampling to 20 Khz. The results were then transfered to the supercomputer and there processed. With a precise planning of all the work that could be done in parallel the otherwise much time-consuming development activities of this project could be carried out in a reasonable time a few weeks. The musical results of this project will be presented in [Ca98A][Ca98B].

Conclusion

The high computational requirements that the modelling of cognitive musical processes imply can be met by the use of supercomputing and parallel technology. Simulations that were inconceivable untill a few years ago are now possible if parallel computers are used. Large scale simulations are generally possible if neural networks models are used. For the SOM, the feed-forward model and other neural paradigms decomposition and parallelization techniques now exist that make the modelling of large cognitive problems possible. On the other side the simultaneous performance of different activities on the same supercomputer and from the same workstation allows a sensible increase in the productivity of the research development cycle. Two examples of distinct approaches to the development of musical cognitive models were presented from the viewpoint of the use of computer technology. The former made use of the parallel computing facilities of the system for solving a highly computationl demanding simulation while the latter facilitated the development work by allowing the simultaneous execution of many tasks related to different problems.

Bibliography

[Le89] Leman M.,Van Renterghem P., *Transputer implementation of the Kohonen feature map for a music recognition task*, Proc. of the 2nd International Transputer Conf.: Transputers for Industrial Applications II (BIRA, Antwerpen Belgium) 1989.

[She64] Shepard R., *Circularity in judgments of relative pitch*, The Journal of the Acoustical Society od America, 36, 2346-2353,1964.

[Kru90] Krumhansl C., *Cognitive foundations of musical pitch*, New York, Y: Oxford Univ. Press.

[Sei91] Seifert U., *The schema concept – a critical review of its development and current use in cognitive science and research on music perception.* In A.Camurri and C.Canepa (Ed.), IX CIM, Genova 1991

[Kra92] Kraaijveld M.,Mao J.,& Jain A., *A non-linear projection method based on Kohonen's topology preserving maps.* In Proceedings of the 11th ICPR. Los Alamitos, CA: IEEE Comput. Soc. Press 1992.

[La92] Langner G., *Periodicity coding in the auditory system*, Hearing Research, 60, 115-142, 1992.

[VIm92] Van Immerseel L.,Martens J., *Pitch and voiced/unvoiced determination with an auditory model.* The Journal of the Acoustical Society of America, 91, 3511-3526,1992.

[Ba94] Balsach L., *Harmonic Convergence*, Translated from La convergiencia Harmonica, Barcelona 1994.

[Ko95] Kohonen T., *Self-Organizing Maps*, 30-Springer Series of Information Sciences, Springer -Verlag, 1995.

[Le95] Leman M., *Music and Schema Theory*, 31-Springer Series in Information Sciences, Springer-Verlag, 1995.

[Ca96] Carreras F.,Leman M., *Distributed parallel architectures for the simulation of cognitive models in a realistic environment.* In E.D'Hollander,G.Joubert,F.Peters,& D.Trystram (Eds), Parallel computing:State-of-the-art perspective. Amsterdam Elsevier, 1996.

[Ca96] Carreras F.,Leman M., *Schema and Gestalt: Explorations of perceptual Learning.* In Proceedings of JIC96-Brugge, M.Leman (Ed.), Brugge 1996

[Le96] Leman M.,Carreras F., *The self-organization of stable perceptual maps in a realistic musical environment.* In G.Assyah (Ed.), Proceedings of the Journees d'Informatique Musicale 1996. Caen: Univ. De Caen – IRCAM.

[Le96] Leman M., Carreras F., *Psychoneural Isomorphism by Computer Simulation.* In Music,Gestalt and Computing , M.Leman (Ed.), Springer LNAI State-of-the-Art Survey 1997

[Ba97] Balsach L., *Application of Virtual Pitch Theory in Music Analysis*, Journal of new Music Research, 26-3, 1997.

[La97] Langner G., *Temporal processing of pitch in the auditory system.* Journal of new Music Research, 26, 1997.

[Le97] Leman M., *The convergence paradigm, ecological modelling and context-dependent pitch perception.* Journal of new Music Research, 26-2,1997.

[Mo97] Moelants D.,Rampazzo C., *A computer system for the automatic detection of perceptual onsets in a musical signal*, In Kansei: the Technology of Emotion, AIMI Ineternational Workshop, A. Camurri (Ed.), Genova 1997

[Ca98A] Carreras F.,Leman M., Petrolino D., *Towards the convergence of the symbolic and subsymbolic approaches to virtual pitch theory*, submitted to Symposium on Musical Cognition and Behavior: Relevance for Music Composing, Rome May 1998.

[Ca98B] Carreras F.,Leman M., Petrolino D., *Content-based extraction of music harmonic information*, submitted to XII Colloquio di Informatica Musicale, Udine Sept. 1998

Previsão de Acordes em Músicas Tonais

Uraquitan Sidney G. C. Cunha e Geber Ramalho

Departamento de Informática -Universidade Federal de Pernambuco
Cx. Postal 7851, 50740-540 Recife, PE, Brazil
{usgcc, glr}@di.ufpe.br

Abstract

What is the real difficult in predict the next chord in one song? What kind of knowledge is necessary to realize this task? All of this questions have been not deeply studied yet, but they are relevant to the development of musical accompaniment systems. The problem of musical chords prediction belongs to a broad and well-known class of problems: *time series prediction*. In our work, we did a comparative analyses between the capacity of prediction of a Neural Network (MLP-Backpropagation) and one symbolic learning algorithm based in decisions trees, the ID3. The results show that a Neural Network has a better performance than the ID3 in all simulations.

1. Introdução

Prever um evento qualquer, normalmente e intuitivamente, requer algum conhecimento prévio do assunto e consequente percepção de padrões de comportamento. Um bom preditor tem que ser capaz de conseguir encontrar os vários padrões existentes dentro do conjunto de treinamento para então conseguir se antecipar e, de fato, prever o que virá. Qualquer aprendizado é bastante limitado a existência de alguma estrutura dentro dos dados. Encontrar esta estrutura é o objetivo do preditor.

Tornar uma máquina capaz de prever alguma coisa, entretanto, é uma tarefa difícil e que deve, sem dúvida alguma, ter boas motivações. Por que prever acordes musicais? Qual a contribuição que este estudo traria para as comunidades de música e da ciência da computação, em especial, da área de Inteligência Artificial? E comercialmente, o que poderíamos ter?

Dentro da comunidade musical, isto seria de uma utilidade surpreendente. Um sistema capaz de prever o próximo acorde de uma música que esteja sendo executada por alguns músicos é um sistema capaz de os acompanhar, ou seja, de tocar algum instrumento junto com estes músicos, o que poderia economizar tempo e dinheiro a muitos deles que dependem de outros músicos, sobretudo para ensaiar.

Como conseqüência, existe uma grande demanda da indústria de software e equipamentos musicais por algo capaz de realizar um bom acompanhamento automático. Hoje, este mercado está povoado de softwares e equipamentos que dão um suporte bastante precário em acompanhamento automático. Os existentes precisam, pelo menos, conhecer previamente a harmonia da música que será tocada (Band in a box). São bastante

previsíveis, estáticos e de pouca utilidade para os bons músicos, sobretudo os músicos que trabalham com improviso.

Dentro da comunidade de computação, resolver um problema deste nível é mais um desafio dentro da área da Inteligência Artificial. O problema de previsão de acordes musicais se encaixa na classe geral de problemas de *time series prediction* (previsão de séries temporais) (Weigend, 1993). O estudo deste problema contribuiria para que esta classe de problemas pudesse ser analisada e estudada detalhadamente através de um cuidadoso trabalho de comparação entre algorítmos de aprendizagem do paradigma conexionista, uma rede neural MLP-backpropagation, e do paradigma simbólico, o ID3.

O objetivo do nosso presente trabalho é a análise de qual ou quais algoritmos melhor se adequam à tarefa de previsão automática de acordes. Na seção 2 falaremos sobre as dificuldades computacionais do nosso problema e sua complexidade e a definição de um acorde musical para a máquina. Na seção 3 falaremos sobre o estado da arte. Na seção 4 será dada uma breve descrição sobre o modelo de rede neural usado e sobre o ID3. Na seção 5 será apresentada a definição que chegamos para um acorde e serão apresentados os resultados que obtivemos em nossos experimentos. A seção 6 concluirá o nosso trabalho, apontando algumas direções para futuros desenvolvimentos.

Dificuldades Computacionais

Prever qualquer coisa, para qualquer ser humano comum, não é uma tarefa trivial. Requer, no mínimo, que se esteja inserido em algum contexto e que se tenha muito conhecimento prévio sobre o assunto. Inserir *contexto* e *conhecimento* dentro de um software é ainda mais difícil. Normalmente não é possível dar à máquina todas as informações possíveis e pertinentes ao assunto para que ela seja capaz de resolver qualquer situação de um determinado problema. Normalmente se define um sub-conjunto do espaço de soluções que melhor represente o espaço total. Este, então, se torna o responsável pela definição de como a solução será alcançada.

O problema de previsão de acordes musicais é bastante complexo sendo, em muitos casos, de difícil solução até para os melhores músicos. Existe uma enorme quantidade de estilos, além da grande variedade de formas de se tocar uma mesma música, o que vai ser muito influenciado pelo estilo de cada músico, pelo que ele esteja querendo expressar naquele momento e até pelo que os outros músicos estejam tocando no momento.

Resolver um problema deste tipo requer uma análise detalhada de uma série de variáveis presentes no contexto musical e que podem ser de função decisiva na implementação do preditor. Encontrá-las e analisá-las é uma tarefa que está intimamente relacionada com a definição do conceito de um acorde musical para a máquina. Como ele será representado para o computador. Basicamente, o acorde pode ser definido como uma *tupla* de algumas destas variáveis mais relevantes a sua caracterização. A definição hoje mais usada é a seguinte:

Acorde = [Tônica, Categoria]

Cada uma destas variáveis (Tônica: C - A - G / Categoria: Maj7 - Min) tem sua função e definição dentro do contexto musical. Sua relevância na formação de um conceito bem

definido de uma acorde musical em música tonal parece ser indiscutível. Com esta definição é possível, até certo ponto, realizar previsão porque existem sequências típicas dentro das músicas tonais, como por exemplo II - V, II - V - I, que facilitam a identificação de padrões dentro da música por parte do preditor. Entretanto, não é muito difícil perceber que esta é uma forma bastante simplificada de definir um acorde. Se analisarmos o problema em maior profundidade percebemos que existem outras variáveis dentro do contexto musical que exercem, às vezes, fundamental influência na definição de qual será o próximo acorde dentro de uma música. Qual então seria a melhor e mais eficaz forma de definir o que venha a ser para a máquina um acorde musical? Que variáveis vão representá-lo?

Dentro de uma sequência musical, além de muitas vezes estarmos a mercê da inspiração do próprio músico que pode mudar a forma de tocar uma música que ele já tocou de uma determinada maneira, existem fatores inerentes a estrutura musical que podem traduzir a forma como o músico prevê o próximo acorde dentro de uma sequência musical. É intuitivo saber que se for tocado um acorde do tipo *Dm7* seguido de um *G7*, ambos com a mesma duração, pode-se concluir com alguma certeza que o próximo acorde será um *C*. Entretanto, se por exemplo o *Dm7* demora o dobro do tempo do *G7*, pode-se achar que o próximo acorde seja um *Db7(#11)*, para que em seguida surja o *C*. Ainda poderíamos ter uma sequência com um *Em7(9)* seguido de *A7*, depois os mesmos *Dm7* e *G7*, todos com a mesma duração. O próximo acorde poderia ser um *Cm7* seguido de um F7 em vez de um *C*. Que conclusão poderíamos tirar desta explanação? Mesmo nas sequências musicais com aparentemente o mesmo padrão de acordes (*Dm7 G7*), temos diferentes acordes seguindo-se a eles. Por estes simples exemplos, podemos perceber a quantidade de variáveis que podem existir dentro da estrutura de uma música e que podem e devem contribuir na definição do conceito de um acorde musical para a máquina. No primeiro caso vemos claramente a importância da variável 'duração'. Quanto um acorde dura naquele momento da música. No segundo caso percebemos a importância das informações de contexto de cada música, ou seja, a importância do histórico de acordes que foram tocados anteriormente dentro da música. Tudo isto contribui, em muitos casos, de forma decisiva para definição de qual será o próximo acorde a ser tocado.

Definir qual será a representação ideal para um acorde, entretanto, não é o único problema que enfrentamos. Partindo desta análise começamos a constatar uma série de dificuldades e problemas que guiaram as nossas decisões sobre a maneira de como tratar o nosso problema. Por exemplo, a tarefa de definir um conjunto de regras que fossem capazes de exprimir e definir qual seria o próximo acorde em uma sequência musical é de extrema dificuldade, pois não existem regras universais dentro da música. O que acontece durante a execução de uma música, quais acordes serão tocados, depende muito do músico, daquilo que ele pode estar sentindo naquele momento e do que os outros músicos estão tocando. Além disto, cada música tem suas próprias peculiaridades, suas próprias características que podem definir esta ou aquela sequência em particular. Os próprios músicos muitas vezes não sabem como responder porque tocaram aquele acorde particular naquele momento, o que traz, logicamente, uma série de dificuldades a respeito de como definir o problema e consequentemente, de como definir regras que pudessem resolver o problema. Daí porque a

melhor solução para este tipo de problema parece ser uma que seja baseada na observação de exemplos, ou seja, baseada em *aprendizado supervisionado*, que é aquele em que os algoritmos recebem padrões de exemplos como entrada e adaptam sua estrutura interna para que a sua saída corresponda a saída desejada (Russel & Norvig, 1996). Por exemplo, poderíamos fornecer como entrada os três últimos acordes de uma música e dizer qual deveria ser o próximo acorde. O algoritmo, com os três acordes fornecidos como entrada, produziria uma saída e a compararia com a que deveria ser, de fato,a saída, que também fora fornecida. Cada vez que a comparação não fosse correta ele readaptaria a sua estrutura interna no sentido de diminuir o erro cometido. O processo de aprendizagem seria composto, então, pela apresentação de um conjunto pré-definido de padrões de exemplos e seria concluído quando a taxa de erros atingisse o seu valor mínimo.

. Uma outra dificuldade presente no nosso problema é a questão da dimensionalidade. Em outros casos exemplo de problemas de previsão de séries temporais (Masters, 1996), como o de previsão de cargas elétricas, de vazão de água em barragens ou de queda ou alta de ações em bolsas de valores, percebemos intuitivamente que é necessário a definição de uma série de variáveis capazes de compor o problema e o aprendizado. Entretanto, em todos estes casos, apenas uma variável será predita. Seja ela a quantidade de cargas, a vazão de águas ou a queda ou alta de ações. Em nosso caso, estamos prevendo qual será o próximo acorde dentro de uma música. Porém, este acorde é formado por uma tupla de não menos que três variáveis (tônica, categoria, duração), o que aumenta a dimensão e a dificuldade do nosso problema, pois o preditor terá que prever não menos que três variáveis.

Um outro problema relacionado com a dimensão é o do tamanho da entrada. Como definir quantos acordes são necessários para que se possa prever o próximo? É importante definir o quanto do passado é importante para que se possa chegar a esta dimensão da entrada.

Estas dificuldades definem o nível do nosso problema e não têm resolução direta, dependendo de um trabalho experimental detalhado. Em nossa abordagem fizemos vários testes com redes neurais e com um algoritmo de aprendizagem baseado em árvores de decisão, mais especificamente ID3.

3. Estado da Arte

Hoje, o único trabalho de que se tem notícia nesta área foi o realizado por Bellinda Thom e equipe da universidade de Carnegie Mellon, (Thom, 1995). Foi desenvolvido um estudo sobre a previsão de acordes em músicas de jazz através da comparação de técnicas de predição e aprendizagem.

A primeira destas técnicas foi chamada de *off line*, e tentava extrair similaridades harmônicas de dentro das músicas de jazz e conseguir com isto capacidade de generalização. Para tanto foi realizado um processo de aprendizado sobre um conjunto estático de canções armazenadas em uma base de dados.

A segunda técnica foi chamada de *on line*, e foi baseada em uma aprendizagem em tempo real, onde cada canção era o seu próprio conjunto de teste e onde o preditor

começava sem qualquer tipo de conhecimento armazenado e a medida que a música ia sendo tocada uma base ia sendo incrementalmente construída. Esta técnica foi testada devida a constatação de que muitas informações eram bastante dependentes de cada música e que, consequentemente, modelos adpatativos seriam bastante necessários.

Por fim, uma última técnica foi testada e chamada de *híbrida*. Ela tentava somar as características fortes do aprendizado usado na técnicas *on line* e *off line* em um único modelo e assim obter melhores resultados.

Seu estudo começa com algumas considerações sobre porque não usar este ou aquele algoritmo de aprendizagem para realizar a previsão de acordes. Devido ao que ela considera uma "falta de uma contínua medida de similaridade" entre cada par de acordes, ela descarta o uso de algoritmos de aprendizagem como as redes neurais, os que usam técnicas de aproximação de funções, técnicas de vizinho mais próximo e algoritmos de aprendizagem discreta que só respondem sim ou não.

Para realizar a aprendizagem, ela usou um algoritmo baseado nos *modelos n-gram* usado pela comunidade de linguagem natural (Bell *et al.*, 1990) e que é capaz de realizar aprendizado incremental e é baseado na probabilidade de acontecer um acorde tendo acontecido já uma determinada sequência anterior de acordes. Em particular, a predição é dada por:

$$Pr(i \mid f_{t+n}, ..., f_t) = Pr(i, f_{t+n}, ..., f_t) / Pr(f_{t+n}, ..., f_t)$$

onde f_t é o acorde mais recente e f_{t+n} é o acorde mais antigo no intervalo de acordes que foram definidos como necessários para que se pudesse prever qual seria o próximo.

Foi definido que um acorde seria representado por sua tônica e sua categoria.

Os testes por ela realizados resultaram em taxas de acerto girando em torno de 42% a 53% entre os modelo off-*line, on-line* e *híbrido*, sendo o modelo híbrido o que teve a melhor performance.

Levando em conta os resultados apenas regulares apresentados pelo presente trabalho, fizemos algumas observações e levantamos algumas críticas e idéias que poderiam vir a melhorar os resultados obtidos se tivessem sido melhor exploradas e que terão um enfoque maior em nosso trabalho:

1. Nenhuma informação sobre a melodia da música no processo de aprendizagem ou teste foi adicionada. Esta informação, sem dúvida, enriqueceria estes processos podendo trazer melhores resultados
2. O corpus de músicas utilizado para o aprendizado e teste foi totalmente transposto para uma única tonalidade, o que diminui o realismo dos testes.
3. A codificação dos acordes é muito pobre. Apenas a tônica e a categoria do acorde foram usadas para definir um acorde musical e já constatamos a importância que uma outra série de variáveis poderia ter na definição de um acorde musical e de como isto poderia influenciar o aprendizado e predição.

4. Nossa Abordagem

O nosso objetivo é basicamente ser capaz de prever um acorde corretamente dentro de uma sequência musical. Para tanto, nos propusemos a realizar uma série de testes com alguns algoritmos de aprendizagem de diferentes paradigmas com o intuito de, primeiro definir qual deles parece ser o que dá os melhores resultados, para em seguida definirmos como trabalhar para dele extrair os melhores resultados.

A idéia foi a de realizar alguns testes com um algoritmo de aprendizagem conexionista, uma rede neural, mais especificamente o modelo Multilayer Perceptron usando o algoritmo de aprendizagem backpropagation (Rumelhart & McClelland, 1986), e um algoritmo de aprendizagem simbólica, o ID3 (Quinlan, 1986), que é baseado em árvores de decisão.

Para o nosso caso de estudo o processo de aprendizado foi realizado a partir de seqüências de acordes de uma base de canções que receberam codificações adequadas para serem passadas para a rede neural e para o ID3. A idéia de como realizar o aprendizado partiu do principio de como o músico percebe e prevê qual será o próximo acorde de uma música que ele não conhece. Normalmente, ele houve dois, três e, às vezes, até quatro acordes para poder ter condições de perceber qual será o próximo. Partindo desta idéia, todo o processo de aprendizado foi guiado através da apresentação de um conjunto de acordes de entrada (dois, três ou quatro) e do fornecimento, para rede neural ou para o ID3, de qual seria o próximo acorde após este conjunto de acordes de entrada (aprendizado supervisionado). Assim, para cada canção que teve as suas sequências de acordes codificadas foram tomados, em principio, por exemplo, os três primeiros acordes como conjunto de entrada e o quarto acorde como o acorde que deveria ser aprendido. Em seguida, tomou-se os três próximos acordes como entrada e aquele que se seguia a eles como saída, e assim sucessivamente como mostra a figura 1.

4.1 O Mulitlayer Perceptron - Backpropagation

O modelo de rede neural escolhido para realizar os primeiros testes no presente problema foi o do *Multilayer Perceptron* usando o algorítmo de backpropagation.

Com forte base matemática o MLP-backpropagation associa unidades básicas de processamento, os neurônios, em camadas que são estimulados por uma determinada entrada, produzindo uma saída cujo valor dependerá dos valores de pesos que estão associados a ligações existentes em cada unidade de processamento. Os valores destes pesos são definidos durante o processo de aprendizagem através do algoritmo backpropagation que os altera de forma que seus valores guardem as informações necessárias ao aprendizado.

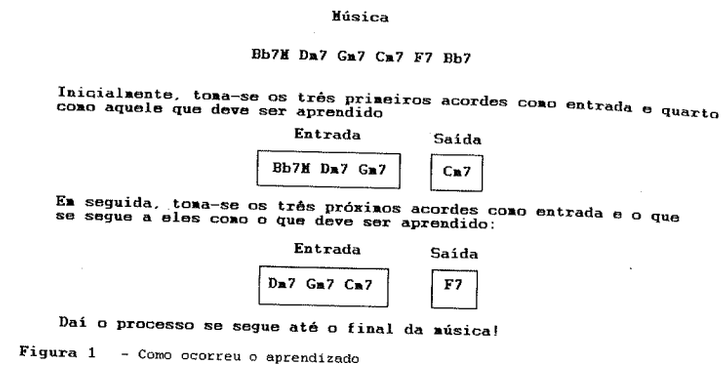


Figura 1 - Como ocorreu o aprendizado

Todo este processo de aprendizado é, normalmente, bastante demorado sendo realizado, às vezes, várias vezes a partir da alteração de parâmetros de configuração da rede que podem definir um novo rumo para o aprendizado e uma melhor ou pior taxa de acertos para a rede. Na figura 2 damos uma idéia de como o processo de aprendizado ocorreu com a rede neural. Neste exemplo, a rede está recebendo como entrada os acordes *Dm7*, *Gm7* e *Cm7* e como saída *F7*.

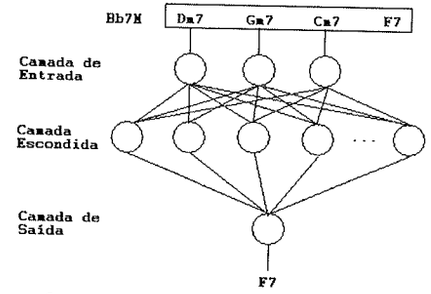


Figura 2 - O Funcionamento da Rede Neural

4.2 ID3

ID3, que significa "Induction of Decision Trees", é um sistema de aprendizado supervisionado que constrói regras de classificação na forma de árvores de decisão. Ele toma um conjunto de objetos, o conjunto de treinamento, como entrada e constrói a árvore de decisão a partir do particionamento deste conjunto. São escolhidos atributos que dividem o conjunto de treinamento e a árvore é construída sobre cada destes sub-conjuntos até que todos os membros de cada um deles pertença a mesma classe.

Uma função heurística é usada para escolher o melhor atributo capaz de separar o conjunto de treinamento. Uma má escolha deste atributo pode afetar o resultado final. A versão original de ID3 tratava apenas valores discretos, sofrendo depois algumas

modificações para tratar também atributos contínuos. Em fase de teste, apenas os atributos de fato relevantes são usados para separar o conjunto. Se estão faltando atributos durante a classificação, todos os possíveis ramos são explorados e a mais provável classificação é escolhida.

ID3 foi desenvolvido por Quinlan e é talvez o mais usado algoritmo ML na comunidade científica e em sistemas comerciais. É uma algoritmo que tem uma alta capacidade de classificação mesmo em conjuntos de dados com muito ruído e tem uma rápida fase de aprendizado com baixa complexidade.

5. Trabalho realizado

O trabalho proposto por Bellinda Thom levanta algumas dúvidas não deixando claro porque exatamente não usar técnicas como as baseadas em redes neurais, ou as baseadas em algoritmos simbólicos, para testar suas capacidades na previsão de acordes. Partindo destas dúvidas e da intuição de que estas técnicas poderiam trazer bons resultados para o nosso trabalho, uma série de testes foram realizados usando a rede neural MLP-backpropagation e o algoritmo de aprendizagem simbólica ID3.

Após uma análise sobre as possíveis informações que poderiam contribuir para a formação e definição de uma acorde musical, chegamos às seguintes variáveis que formaram a tupla que definiu o acorde musical usado em nossos experimentos:

[Tônica, Categoria, Intervalo, Posição no Compasso, Posição na Música, Duração]

Cada uma destas variáveis representa características bem definidas de cada acorde. A seguir definimos cada uma delas:

- Tônica: É o que define qual acorde será tocado. Ex: C, D, E, etc;
- Categoria: É o tipo do acorde que será tocado: Ex.: min, maj7, etc;
- Intervalo: é o intervalo que separa o presente acorde do anterior a ele. Ex.: min, maj, etc
- Posição no compasso: Variável que indica a posição do acorde dentro do compasso em que ele se encontra na música;
- Posição na música: Variável que define a posição do acorde dentro da música. Ex.: Begin, End, Turnround, etc;
- Duração: Variável que representa a duração do acorde. Ex: 1,2 3,4, etc.

Usando um corpus formado por 58 canções de jazz e usando a definição acima para representar cada acorde, realizamos uma série de testes e obtivemos os seguintes resultados:

5.1. Resultados obtidos com a Rede Neural:

Foram realizados testes com treinamento usando dois, três e quatro acordes de entrada para a rede tentar prever qual seria o próximo. Definiu-se que, aproximadamente 75% da base de acordes criada constituiria o conjunto de treinamento da rede e que, aproximadamente, 25% da base de canções seria usada para formar o conjunto de testes. O

software utilizado para realizar os treinamentos e testes foi o Qnet97 que possui um mecanismo interno que possibilitou a realização de validação cruzada automaticamente.

Nos primeiros testes, expostos na Tabela 1, a rede foi treinada objetivando a previsão de todas as variáveis que compunham a tupla de um acorde. Eis alguns dos melhores resultados obtidos:

Número de Acordes de Entrada	Número de Unidades na camada escondida da rede	Número de interações de treinamento	Erro médio no conjunto de teste	Erro médio no conjunto de treinamento
2	10	2600	19.4%	19.0%
3	10	12000	16.8%	14.2%
4	10	2000	19.5%	19.0%

Tabela 1: Previsão de todas as variáveis de formação do acorde

Em seguida, foram realizados uma série de testes com dois e três acordes de entrada para a rede, com o aprendizado objetivando a previsão de um acorde formado apenas pela tônica e pela sua categoria, pois estas duas informações resolveriam grande parte dos nossos problemas. Os melhores resultados estão expostos na Tabela 2.

Número de Acordes de Entrada	Número de Unidades na camada escondida da rede	Número de interações de treinamento	Erro médio no conjunto de teste	Erro médio no conjunto de treinamento
2	10	10000	16.3%	11.2%
3	10	10000	15.4%	11.5%

Tabela 2: Previsão de Tônica e Categoria

5.2 Resultados obtidos com ID3:

Foram realizados testes na mesma base de canções, objetivando a previsão, em principio, apenas da categoria do acorde (Tabela 3), depois da tônica e categoria do acorde (Tabela 4).

Com validação cruzada	Usando 25% do conjunto de pares de entrada e saída para testes	Erro médio
Não	Não	13.4%
Sim	Não	23.2%
Não	Sim	24.8%
Sim	Sim	25.7%

Tabela 3: Previsão de Tônica

Com validação cruzada	Usando 25% do conjunto de pares de entrada e saída para testes	Erro médio
Não	Não	23.7%
Sim	Não	44.8%
Não	Sim	49.8%
Sim	Sim	48.0%

Tabela 4: Previsão de Tônica e Categoria

6. Conclusão

Objetivamos o desenvolvimento de algo pioneiro dentro da área da computação aplicada à música, haja visto que ninguém publicamente tratou este problema tão detalhadamente. Diante dos nossos primeiros resultados já concluimos que o sistema de fato é factível e que, partindo da idéia de que usamos praticamente o estilo musical de maior complexidade harmônica, que é o jazz, em todos os nossos testes, supõe-se que obteremos resultados bem melhores em estilos mais simples e populares.

Apesar de não termos usado o mesmo corpus que o utilizado por Thom em seu trabalho, estes primeiros testes que realizamos mostram resultados bem superiores aos resultados obtidos em seu trabalho.

Apesar destes bons resultados, principalmente com as redes neurais que parecem ser o melhor caminho para a resolução deste problema, uma série de outros testes ainda serão realizados, principalmente para busca da melhor forma de representar o acorde. Existem outros aspectos e variáveis que em muitos casos são fundamentais na decisão de qual acorde tocar. Variáveis como a tonalidade local de uma música e a própria melodia são intuitivamente de grande importância e serão objeto do nosso próximo passo de análises e estudos.

7. Referências

Bell T. C., Cleary J. G. & Witten I. H. (1990). *Text Compression*. Prentice Hall Advanced References Series, Prentice Hall, New Jersey.

Masters T. (1996). Neural, Novel and Hybrid Algorithms for Time Series Prediction. Wiley.

Quinlan, J. R. (1986). Induction of Decision Trees. Machine Learning, 1:81-106.

Rumelhart D.E. & McClelland J.L. (1986). Parallel Distributed Processing, volume 1. MIT Press.

Russel S. & Norvig P. (1996). Artificial Intelligence: A Modern Approach. Prentice Hall, Englewood Cliffs New Jersey 07632

Thom B. (1995). Predicting Chordal Transitions in Jazz: The Good, the Bad, and the Ugly. 14th International Joint Conference on Artificial Intelligence (IJCAI-95) Montréal, Canada

Weigend, A. S. (1993). Time Series Prediction; Forecasting The Future And *Understanding The Past.* Addison Wesley Pub Co. Volume 0015.

Controle Paramétrico MIDI Usando Interface Gestual Ultrassônica

FURIO DAMIANI [1,2], JÔNATAS MANZOLLI [2] E GILBERTO MENDES [1]

UNICAMP - Universidade Estadual de Campinas / 13081-970 Campinas, SP Brasil

furio@fee.unicamp.br, jonatas@nics.unicamp.br

[1]*Faculdade de Engenharia Elétrica e de Computação*

[2]*Núcleo Interdisciplinar de Comunicação Sonora*

Abstract

An application of an ultrassonic interface (IGu), developed in the Gestual Interfaces Laboratory at NICS is presented. A short description of the IGu is given. The parametric MIDI control developed is described an well as its functionality. The IGu has a small ultrassonic transmitter, worn by the user in one finger, sending signals to a sensor array. The finger position is sampled 80 times per second, generating a stream of coordinates (x,y,z) that are fed to the parametric MIDI control program. A set of MIDI parameters controlled by the coordinates values was established, in order to give interesting sound output capabilities. The control program uses a graphic interface to ease the system's utilization.

INTRODUÇÃO

O trabalho interdisciplinar aqui apresentado faz parte dos objetivos do Laboratório de Interfaces Gestuais, na linha de estudo de sistemas para o sensoreamento gestual e de software para utilização em performances de produção musical interativa.

Tem havido um interesse crescente em Interfaces Gestuais para uso em situações de performance. Nos mais recentes SBC&M foram apresentados trabalhos sobre composição interativa usando interfaces de diversos tipos [Iazetta 1994; Kuntenback & Hulteen 1993; Lima & Wanderley 1996; Mulder 1994; Manzolli 1995]. Estes dispositivos permitem efetivar propostas de interação onde o movimento do *performer* é um dos vetores geradores da composição musical.

Seguem-se a descrição da Interface Gestual Ultrassônica (IGu) e do software que implementa o controlador paramétrico MIDI, que utiliza as coordenados gestuais fornecidas pela IGu.

INTERFACE GESTUAL ULTRASSÔNICA

Foi projetado e construído um dispositivo de entrada que fornece o posicionamento instantâneo de um dedo da mão do usuário [Damiani & Mendes 1998]. A trajetória de um gesto é amostrada ponto a ponto, pré-processada e fornecida ao programa de controle paramétrico MIDI.

É utilizado um transmissor de ultra-som fixado ao dedo, que aponta para um conjunto de sensores. São transmitidos, em fase, sinais idênticos de ultra-som e infra-vermelho. Este último tem tempo de propagação desprezível se comparado ao do ultra-som, sendo detectado junto ao conjunto-sensor para marcar o início da transmissão (como um sinal de sincronismo).

Os sensores de ultra-som detectam o instante de chegada da onda esférica, usando a aproximação do transmissor como sendo pontual (Fig 1). Desta forma, o sinal de infra-vermelho recebido determina o inicio das temporizações de propagação, e cada sinal de ultra-som detectado determina o fim de uma temporização, resultando na medida de tempos com os quais se calculam as distâncias [Nonaka & Da-te 1995], usando a equação :

$$d = v \, . \, t \qquad \text{[metros]}$$

onde d é a distância, v é a velocidade do som e t o tempo transcorrido.

O cálculo das coordenadas x y z do transmissor se faz considerando cada receptor como centro de uma esfera e as três distâncias como raios dessas esferas. O sistema de 3 equações de esferas tem como incógnitas as coordenadas x y z do transmissor. A interseção das 3 esferas será a solução única no semi-espaço considerado.

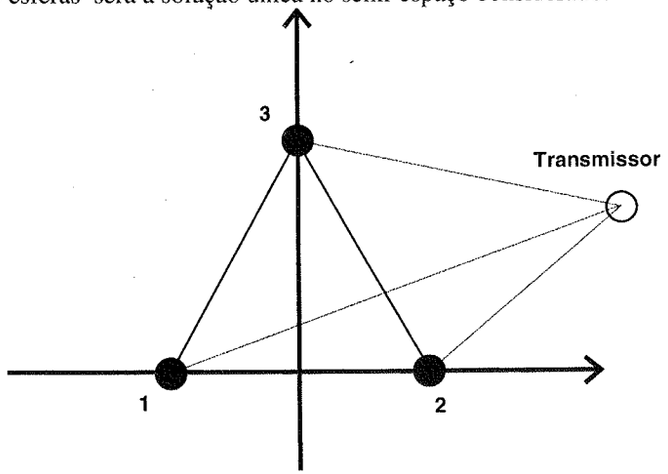


Fig. 1 Disposição dos receptores em relação ao transmissor.

Os sinais de ultra-som são detectados quando sua amplitude ultrapassa um limiar [Barshan 1992; Parilla, Anaya & Fritsch 1991]. A resolução em cada medida de distância, de aproximadamente 5 milímetros, permite uma diferenciação satisfatória de posição mesmo em movimentos à distância máxima, determinada pela sensibilidade do circuito, de cerca de 3 metros. Em favor da simplicidade, foram evitadas análises mais complexas dos sinais, como a do seu conteúdo de freqüências, ou da dinâmica de sua envoltória [Datum, Palmiere & Moiseff 1996; Sabatini 1995]. As imprecisões advindas desta forma de detecção não comprometem o resultado final, pois o pré-processamento dos dados de um gesto compensa os erros absolutos.

A atenuação do sinal ultra-sônico propagando-se no ar se dá de forma exponencial; portanto, o circuito de condicionamento de sinais de um receptor deve lidar com uma ampla faixa de amplitudes. Para executar esta tarefa, foi usado um *Schmitt Trigger* com um limiar que se adapta continuamente, proporcional ao nível de sinal recebido. Além disso, outros circuitos são necessários para aumentar a relação sinal-ruído, como podemos ver no diagrama de blocos abaixo, constando de um pré-amplificador de ganho alto na entrada, um filtro passa-faixa centrado em 40 Khz, um detetor de envoltória e o detetor de limiar variável. Na saída, vemos um conversor para tensões de 0 a 5 volts, apropriadas para o bloco digital que será mencionado mais adiante.

Para compensação da variação da velocidade do som com a temperatura, foi incluído um sensor de temperatura com saída linear em tensão.

A avaliação de posição instantânea do (dedo) transmissor é feita, no sistema implementado, a uma freqüência de aproximadamente 80 vezes por segundo. O tempo de propagação do sinal de ultra-som é medido em 3 contadores digitais, implementados em uma EPLD de 68 pinos. Cada contador é inicializado no início da transmissão (identificado pela recepção do sinal de infra-vermelho) e a contagem é interrompida no início da recepção de cada sensor de ultra-som, de forma independente. Os valores armazenados nos contadores são lidos pela interface paralela padrão Centronics de um PC-compatível. São endereçados pela saída de dados da mesma e lidos através dos bits de status, transferidos de 4 em 4 bits, num total de 64 bits.

A integração de praticamente toda a parte puramente digital do circuito em um só circuito integrado permitiu a confecção de um dispositivo compacto e de baixo custo, características essenciais para torná-lo útil como ferramenta de desenvolvimento.

Fig. 2 Diagrama de blocos do sistema

CONTROLE MIDI

Nossa pesquisa em interfaces gráficas para composição algorítmica [Manzolli & Ohtisuki 1996] foi o ponto de partida para o desenvolvimento deste controle paramétrico

MIDI. A idéia básica foi tornar o mapeamento das coordenadas (x,y,z) fornecidas pela interface ultrassônica o mais flexível possível. Foram criados dois módulos de contrôle: Controle por Faixas MIDI (CPFM) e Controle de buffer circular MIDI (CBCM).

No CPFM as coordenadas são mapeadas diretamente a eventos MIDI. O valor da coordenada z é tomado como o *Status Byte* e os valores de x e y como os dois *Data Bytes* da mensagem MIDI gerada.

Com isto criam-se regiões de controle delimitadas por planos, situadas à frente do usuário. Cada região é associada a uma classe de eventos diferente dando ao usuário a possibilidade de escolher as regiões de geração de *Midi Notes, Control Change* e *Program Change* (Tab.1). Ao fazer movimentos com a mão, o usuário fará variar as coordenadas e, portanto, variarão os parametros MIDI de acordo com a o contrôle paramétrico estabelecido.

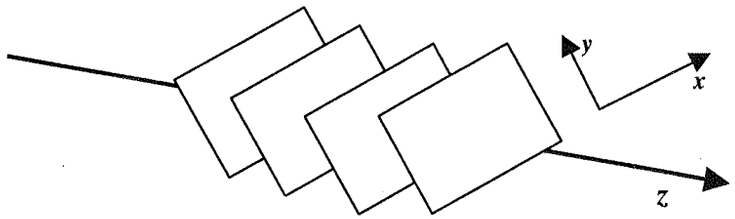


Fig. 3 Controle Paramétrico por Faixas MIDI

Tab. 1 Relação entre coordenadas e parâmetros MIDI.

Status	**Limites da região de controle** (metros)	***Byte 1*** *(y)*	***Byte 2*** *(x)*
Note On	$2,0 < z < 1,7$	[1..127]	[1..127]
Pitch Bend	$1,7 < z < 1,5$	64	[1..127]
Control Change (Volume)	$1,5 < z < 1,4$	01	[1..127]
Control Change (Pan)	$1,4 < z < 1,3$	11	[1..127]
Program Change	$1,3 < z < 1,2$	[0..127]	0

No CBCM estabelece-se um processo de modificação em eventos MIDI previamente armazenados num *buffer* circular. Os valores das coordenadas *(x,y,z)* são utilizados por operadores para gerar essas modificações sonoras na seqüência original. Foi estudado um elenco de modificações com sentido musical. Os operadores usam um coeficiente de combinação percentual que varia entre [0..1] e cada evento modificado é armazenado na mesma posição que ocupava no *buffer*, gerando-se assim um ciclo de transformações sonoras.

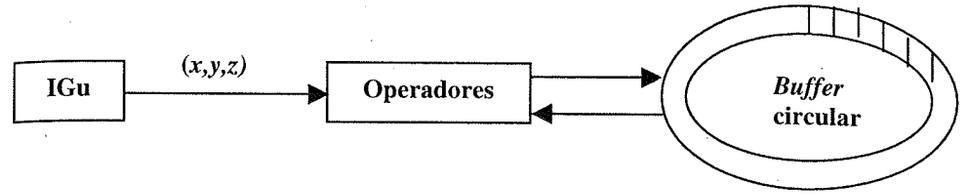


Fig. 4 Controle de *buffer* circular MIDI

Os operadores implementados usam a norma da distância das coordenadas (x,y,z) à origem (0,0,0) para controlar suas operações:

$$\text{Transposição} = \text{MIDI}value + Sgn\,(x\text{-}y)*Sqr(x^2 + y^2 + z^2)$$

$$\text{Combinação} = \alpha\ \text{MIDI}value + (1\text{-}\alpha) * Sgn\,(x\text{-}y)*Sqr(x^2 + y^2 + z^2)$$

onde *Sgn* é a função sinal, *Sqr* é a raiz quadrada e $0 < \alpha < 1$.

Os dois módulos podem ser acoplados; então, ao mesmo tempo em que se criam eventos MIDI através de CBCM os mesmos podem ser combinados com os eventos existentes no *buffer*.

CONCLUSÃO

Novos dispositivos surgem como uma proposta para a interação sonoro-visual, onde o gesto do intérprete integra-se ao som produzido. Para este temos três aplicações em vista: obras com tape e interface gestual, obras com solo de interface gestual e, finalmente, obras interativas com computador e interface gestual.

Pretendemos estender o uso da IGu para controle de equipamento multimídia (iluminação, vídeo projetores, etc). Iremos estudar sistemas de identificação gestual com redes neurais artificiais e desenvolveremos novos métodos de interação entre as coordenadas (x,y,z) e processos sonoros gerados em computador, como, por exemplo, processos de síntese de som.

REFERÊNCIAS

Barshan B. (1992) A Bat-Like Sonar System for Obstacle Localization, *IEEE Transactions on Systems, Man, and Cybernetics,* 22(4), 636-647.

Damiani, F. & Mendes, G. (1998) Interface gestual ultrassônica. *IV Workshop Iberchip,*

Datum, M. S., Palmiere, F. & Moiseff A. (1996) An Artifical Neural Network for Sound Localization Using Binaural Cues, *Journal of the Acoustic Society of America,* 100(1), 372-383.

Iazzetta, F. (1994). Um Novo Músico Chamado Usuário. *I Simpósio Brasileiro de Computação e Música*, 231-235.

Kuntenback, G. & Hulteen E. A. (1993) Gestures in Human-Computer Communication. *The Art of Human-Computer Interface*. Addison-Wesley

Lima, G.H.T. & Wanderley, M. (1996) Dance-music interface based on ultrasound sensors and computers, *III Simpósio Brasileiro de Computação e Música*, 12-16.

Manzolli, J. (1995) The development of a gesture interface laboratory. *II simpósio brasileiro de computação e música*, 81-84.

Manzolli, J. & Ohtisuki, W. (1996). Interasom: Um Desktop Para Composição. *III Simpósio Brasileiro de Computação e Música*, 84-88.

Mulder, A (1994). Virtual musical instruments: Acessing the sound synthesis universe as a performer. *I Simpósio Brasileiro de Computação e Música*, 243-250.

Nonaka, H. & Da-te T. (1995) Ultrasonic Position Measurement and Its Application to Human Interface, *IEEE Transactions on Instrumentation and Measurement*. 44(3), 771-774.

Parrilla, M., Anaya, J.J. & Fritsch, C. (1991) Digital signal processing techniques for high accuracy ultrasonic range measurements, *IEEE Transactions on Instrumentation ad Measurement,* 40(4), 759-763.

Sabatini A.M. (1995) A Digital Signal-processing Technique for Compensating Ultrasonic Sensors, *IEEE Transactions on Instrumentation ad Measurement,* 44(4), 869-874.

AGRADECIMENTOS

Agradecemos aos membros do NICS pela discussões durante nossos seminários de Música e Tecnologia. Este trabalho teve o apoio da FAPESP e do CNPq.

Análises Melódicas usando Grafos

RODRIGO PERES FRANCO FURTADO
HUMBERTO JOSÉ LONGO
Instituto de Informática - UFG
Caixa Postal 131, Goiânia - GO, Brasil
CEP: 74001-970

Abstract

This paper presents an approach to represent keys (modulations) by a graph structure and how to use it to help melodic analisis based on musical rules of harmonization and composition. The main objetive is just to show that the graph teory contains important elements that would be used, with a certain facility, as support in melodic analisis.
Key words: Melodic analisis, keys (modulations), graph.

1. Introdução

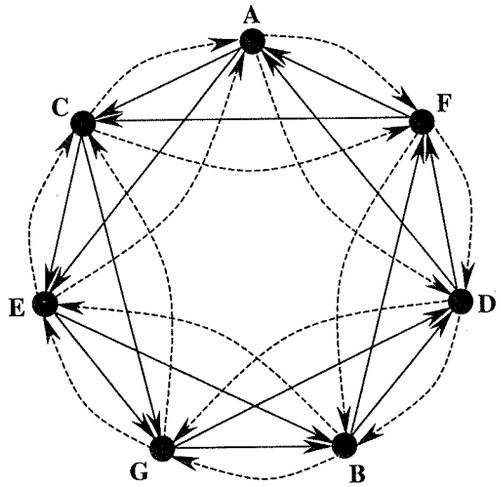
O objetivo geral deste estudo é mostrar como conceitos básicos da teoria musical e da teoria de grafos ((Jurafsky 1971), (Priolli 1987) e (Watkins & Wilson 1990)) podem se relacionar, permitindo a criação de métodos de análise melódica que visem uma correta harmonização musical. Logicamente, este artigo não dá uma cobertura total da análise melódica e harmonização, o que exigiria um estudo muito mais aprofundado. Este estudo se detém na análise teórica das partes melódicas, usando recursos da teoria de grafos e logicamente obedecendo a algumas regras musicais de harmonização baseadas em tríades e encadeamentos de acordes. Um campo harmônico pode ter notas com acidentes (sustenidos e bemóis) que em nada atrapalham a utilização da representação proposta para o campo.

A análise musical aqui realizada é bastante restrita, porém, é o ponto de partida para outras análises mais complexas. Aqui é levado em conta apenas a análise das tríades em compassos simples, com notas de mesma duração e utilizando-se somente acordes consonantes. Para facilitar considerou-se que cada compasso pode ter um e somente um acorde. A questão do encadeamento de acordes será aqui tratada de forma também simplificada, seguindo uma regra específica e bem definida.

2. Modelagem do campo harmônico de C maior

Na aplicação da teoria de grafos à música, o que foi feito a principio foi o estudo de uma possível modelagem dos campos harmônicos com grafos e como utilizá-los para analisar as tríades dos campos harmônicos. Neste estudo é apresentado apenas a modelagem do campo harmônico de C maior, contudo os demais campos harmônicos têm modelagem similar.

Assim, seja o grafo $G = (VG, AG\_asc \cup AG\_des)$. O conjunto de vértices VG do grafo G corresponde as notas naturais que formam o campo harmônico de C. O conjunto de arestas $AG = AG\_asc \cup AG\_des$ contém arestas dos tipos ascendentes e descendentes. Visto que cada acorde é formado por três notas, as arestas descendentes representam a ligação (formação do acorde) entre estas três notas. As arestas são dirigidas de cada tônica para sua 3ª e 5ª (de um vértice pai para os seus vértices filhos), para que se possa encontrar todas as tríades do campo harmônico. As arestas ascendentes ligam uma nota às suas possíveis tônicas (arestas dirigidas do vértice filho aos possíveis vértices pai). Na ilustração a seguir, tal grafo é representado com as arestas descendentes desenhadas em linha cheia e as ascendentes em linha tracejada:



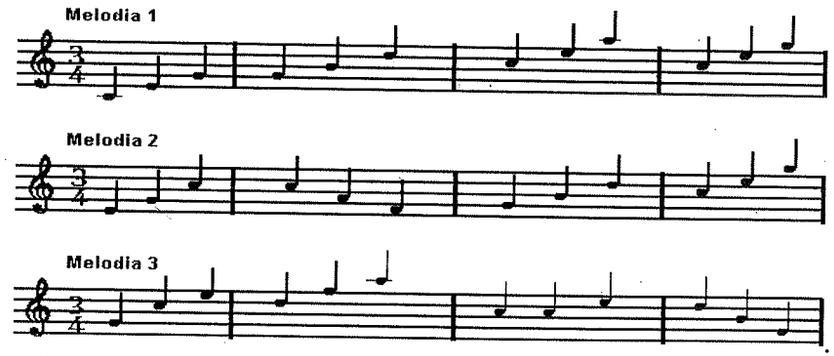
Com esta modelagem do campo harmônico de C, pode-se determinar, compasso a compasso, as possíveis tríades de uma melodia com um percurso apropriado no grafo. No algoritmo descrito a seguir, parte-se da primeira nota de cada compasso e busca-se a mesma no grafo. A partir daí, faz-se uma busca em largura modificada (*breadth first search*, (McHugh 1991)) no grafo, comparando-se os vértices adjacentes ao vértice de entrada (primeira nota do compasso) com as outras notas do compasso. Consideram-se inicialmente os vértices alcançáveis a partir das arestas descendentes e depois os vértices alcançáveis a partir das arestas ascendentes. A seguir será mostrado como encontrar os elementos de uma tríade e julgá-los adequados ou não para uma possível harmonização.

O algoritmo de análise aqui proposto inicialmente encontra todas as combinações das notas de cada compasso (a ordem das mesmas não interessa). As combinações serão do tipo $C_{comp,3}$, onde *comp* é o número de notas do compasso, pois busca-se a validação de uma possível tríade. A seguir comparam-se estas combinações com o resultado de um percurso no grafo (busca em largura). Para esta análise, foi adotado fixar a primeira nota da combinação e a partir das outras notas procurar tríades completas ou possíveis fragmentos. Portanto, uma vez determinado que o campo harmônico é de C, os seguintes passos devem ser executados:

1 - Gerar as combinações de notas do compasso;
2 - Fazer a seguinte verificação nas combinações de notas:
3 - Iniciar o percurso no grafo começando pela 1ª nota de cada combinação;
4 - Se existir arestas da 1ª nota da combinação para as duas outras notas do compasso
5 - Achou-se uma tríade completa;
6 - Senão
7 - Se essa 1ª nota tiver um pai comum com alguma das notas do compasso
8 - Achou-se uma tríade válida (fragmento de tríade);
9 - Senão
10 - Não existe tríade válida para a combinação.

3. Exemplo de Análise

A seguir é mostrada a análise e harmonização das melodias abaixo, usando-se o grafo e o algoritmo descritos anteriormente. Contudo, é importante ressaltar que no algoritmo descrito anteriormente a análise é feita para todas as combinações de notas (tomadas três a três) de cada compasso. Nos três exemplos descritos a seguir não se verificou todas as combinações, pois pretende-se com os exemplos apenas se mostrar que o grafo proposto adequa-se à análise das tríades dos compassos de uma melodia.



Melodia 1:

1º compasso : Observa-se claramente a existência de uma tríade de C (dó, mi, sol), concluindo-se então que o acorde que melhor se ajusta é C maior. No

grafo, começa-se a busca pelo vértice C e verifica-se se algum dos vértices adjacentes ao vértice C corresponde a alguma das notas do compasso. Como em ambos os casos (para os dois filhos) a resposta é afirmativa, tem-se uma tríade completa de C, no estado fundamental.

2º compasso : Aqui há uma tríade completa de G (sol, si, ré), o que justifica o uso de G para harmonizar este compasso. A busca no grafo é semelhante à feita anteriormente e como as notas seguintes, após o sol, são filhos de G, então esta é a melhor tríade.

Para o terceiro e quarto compassos a análise é semelhante as duas dadas anteriormente.

Melodia 2:

1º compasso : Neste compasso, observa-se que as três notas formam uma tríade de C (1ª inversão), portanto a harmonia mais justa é o acorde de C. No grafo, a primeira nota do compasso leva ao vértice E e a segunda nota leva ao filho de E, vértice G. Neste ponto, vale observar que tem-se um fragmento de tríade (E,G). Para saber se existe uma tríade melhor (completa), observa-se o outro filho de E, como este não corresponde à terceira nota do compasso, observa-se, então, os pais das notas do fragmento de tríade (E,G). Se o pai de G for igual ao pai de E então tem-se uma tríade completa, cuja tônica é C e a 3ª e 5ª são E e G, respectivamente. Portanto, a melhor tríade é exatamente esta última.

Neste 1º compasso percebe-se bem a utilidade dos apontadores para os pais das notas, visto que, sem estes apontadores teríamos uma harmonia correta, mas que não seria a melhor possível, pois a última nota do compasso (C) ficaria praticamente isolada. Este isolamento não acontece com o grafo proposto, pois, sempre que se quiser saber se existe uma tríade que ajusta-se melhor, basta verificar o pai das notas do compasso (que já foi analisado) e comparar o resultado com a(s) outra(s) nota(s) do compasso, para verificar se existe uma tríade completa.

2º compasso : Este compasso apresenta uma tríade que não é nem a primeira nem a segunda inversão, porém possui as três notas da tríade de F. No grafo, inicia-se a busca pelo vértice C e verifica-se logo que a tríade procurada não está ali, pois a segunda e terceira nota não são filhos diretos de C. Nota-se também, que a segunda nota do compasso não tem correspondência direta com C. A alternativa é procurar o pai de C (terceira nota do compasso, vértice F), formando uma tríade válida (fragmento de tríade). Pode-se, ainda, verificar se existe uma tríade completa. Neste caso, observa-se que outro filho de F (vértice A), é a segunda nota do compasso. Conclui-se, portanto, que se trata de uma tríade de F.

No terceiro e quarto compassos a análise é direta, visto que as tríades estão no estado fundamental e pode-se achar o acorde diretamente por uma verificação simples .

Melodia 3:

1º compasso : Trata-se claramente de uma tríade (2ª inversão) de C. No grafo, começa-se no vértice G, (1ª nota do compasso); como a segunda e a terceira nota do compasso não são filhos de G, conclui-se que G é parte de alguma tríade. Busca-se, então, um dos pais de G, por exemplo C. Imediatamente já se sabe que C é a segunda nota do compasso. Analisando-se agora o outro filho de C (vértice E) encontra-se a tríade completa, pois a terceira nota é E (que é a terça da tríade de C).

2º compasso : Nota-se uma tríade de Dm, o que justifica o uso deste acorde neste compasso. No grafo a análise é direta, chegando-se a conclusão que é uma tríade de Dm .

3º compasso : Neste compasso nota-se que as duas primeiras notas são iguais, o que não causa nenhum problema, pois pode-se insistir em qualquer nota da tríade sem mudar a harmonia. Nota-se ainda que a terceira nota faz parte da tríade de C (mi), e portanto harmoniza-se este compasso com C. No grafo, analisa-se o primeiro compasso e chega-se à conclusão que existe neste compasso um fragmento de tríade, e assim escolhe-se a tríade de C.

4º compasso : Neste compasso há uma tríade completa de G, portanto o melhor acorde é G. No percurso no grafo vai-se diretamente ao vértice D, analisam-se os seus filhos comparando-os com as notas (2ª e 3ª) do compasso. Como as notas não correspondem a nenhum dos filhos de D, busca-se o pai de D. Como o pai de D é também uma nota do compasso, já se encontrou uma tríade válida. Para verificar se há uma tríade melhor, examina-se o outro filho de D (vértice G), e verifica-se que ele também faz parte das notas do compasso, e encontra-se, portanto, uma tríade completa.

Observe-se que neste último compasso usou-se o recurso de, sempre que uma nota do compasso (principalmente a primeira) não tiver um filho correspondente às outras notas do compasso, se buscar o pai desta nota e verificar se ele ou o outro filho correspondem as outras notas do compasso.

4. Encadeamento de acordes

O encadeamento entre acordes também é uma parte importante a ser observada para uma correta análise musical. Neste estudo, juntamente com a análise de tríades, iniciou-se o estudo dos encadeamentos como forma de complementar a análise musical. É lógico que a composição musical não é algo tão inflexível e exato para que possamos simplesmente ditá-las por regras e nada mais. Porém, pode-se arriscar uma tentativa de cobrir alguns aspectos que possam ser modelados por regras e conceitos que não ceifem a criatividade artística.

Neste estudo, após a observação das tríades para cada compasso, aplicou-se uma regra básica de encadeamento de acordes, compasso a compasso, gerando todas as possíveis harmonias para uma dada melodia. Assim, usou-se o critério de só permitir encadeamentos

em que os acordes que estão se encadeando tenham pelo menos uma nota em comum; salvo o caso em que os acordes representem funções principais (IV e V graus).

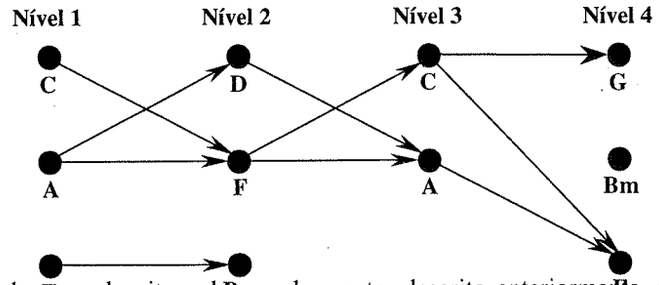
O algoritmo descrito anteriormente produz como resultado os possíveis acordes para cada compasso de uma melodia. Para se descobrir quais são os possíveis encadeamentos para a melodia, usou-se um segundo grafo ($G_1 = (VG_1, AG_1)$) para modelar estes encadeamentos entre os compassos. Neste grafo, o conjunto de vértices VG_1 é formado pelos acordes de cada compasso. Esses vértices podem ser representados em níveis (um nível para cada compasso), ficando os acordes do primeiro compasso no primeiro nível, os acordes do segundo compasso no segundo nível e assim sucessivamente. O conjunto de arestas é formado por possíveis encadeamentos entre dois acordes de níveis sucessivos. Assim, fazendo-se buscas em profundidade (*depth first search*, (McHugh 1991)) a partir dos vértices do primeiro nível, torna-se possível listar todos os possíveis encadeamentos de todas os acordes, ou seja, todas as possíveis harmonias para a melodia.

Um algoritmo simples para a análise harmônica pode ser descrito como a seguinte seqüência de operações:

1 - verificar a armadura de clave da música e identificar qual é o seu campo harmônico;
2 - analisar compasso a compasso, identificando-se as tríades existentes;
3 - usar os critérios descritos na seção 2 para gerar opções de harmonia para cada compasso;
4 - determinar os possíveis encadeamentos entre os compassos.

Embora seja um algoritmo genérico, o passo 3 pode corresponder aos passos 3 a 10 do algoritmo descrito na seção 2 e o passo 4 pode corresponder ao algoritmo de encadeamento descrito nesta seção.

A seguir é mostrado um exemplo de grafo de encadeamento gerado para a melodia 3 da seção anterior. Verifica-se que as arestas ligam acordes que possuem pelo menos uma nota em comum:



Aplicando-se o algoritmo de encadeamento, descrito anteriormente, a este grafo são geradas as seguintes possibilidades de harmonia:

a) C, F, C, G;
b) C, F, C, Em;
c) C, F, Am, Em;
d) Am, Dm, Am, Em;
e) Am, F, C, G;

f) Am, F, C, Em; e
g) Am, F, Am, E.

O conjunto de possíveis harmonias geradas por este algoritmo pode ser muito grande. Contudo, este conjunto pode ser restringido usando-se outras regras de harmonização juntamente com a regra utilizada (por exemplo, cadência plagal e cadência perfeita).

5. Conclusão

Observou-se, claramente, neste estudo como áreas que parecem tão distantes podem ser relacionadas e contribuírem mutuamente para seus desenvolvimentos. Os aspectos musicais têm fortes marcas de lógica e simetria, tornando possível gerar harmonias com o uso de estruturas de dados e algoritmos que obedeçam a determinadas regras musicais. Com os algoritmos apresentados pode-se ter uma idéia mais clara de que realmente é possível se criar métodos para o auxilio na análise de uma melodia.

O que foi feito até agora porém é uma fração muito pequena do que ainda pode ser estudado nesta área. Apesar de ser é um estudo inicial, mostrou-se aqui que é viável a proposição de algoritmos mais avançados para cobrir um número maior de casos e proporcionar uma harmonização mais correta para uma melodia.

Referências

Jurafsky, A. (1971). *Manual de Armonia*. Buenos Aires: Ricordi Americana.

McHugh, J. A. (1991). *Algorithmic Graph Theory*. London: Wiley.

Priolli, M. L. M. (1987). *Harmonia da concepção básica à expressão contemporânea, 1º volume*. Rio de Janeiro, RJ: Casa Oliveira de Música Ltda.

Watkins, J. J. & Wilson, K. J. (1990). *Graphs. An Introductory Approach*. New York: Wiley.

Internet Music: Dream or (virtual) Reality?

FABIO KON
f-kon@cs.uiuc.edu
Department of Computer Science
University of Illinois at Urbana-Champaign
1304 W. Springfield Avenue.
Urbana, IL 61801 — USA

FERNANDO IAZZETTA
iazzetta@exatas.pucsp.br
Laboratório de Linguagens Sonoras
Comunicação e Semiótica — PUC-SP
R. Ministro Godoy, 969 - sala 4B-06
São Paulo - SP 05015-901 — Brasil

Abstract

The recent explosive growth of the Internet and the fact that personal computers with multimedia capabilities are now a commonplace have raised the interest in distributed multimedia systems. Among all multimedia applications, the ones that relate to interactive music are the ones which present the tightest timing constraints. Traditional operating systems and networking infrastructure does not provide enough support for the quality of service these applications require.

This paper describes a number of problems concerning distributed musical applications on the Internet and discusses possible solutions for each of them. We, then, consider a variety of situations in which Internet Music systems could be used and suggest that the Internet is a viable environment for musical activities even when using existing technologies.

1 Introduction

In 1995, our research group at the University of Illinois developed *Vosaic*, an extension to the Mosaic Web Browser that is capable of streaming compressed video

and audio in real time over the Internet using adaptive algorithms [Chen et al., 1996, Tan et al., 1996]. This pioneering work demonstrated that it was possible to send complex multimedia data over the Internet in real-time. However, it also showed that the complete unpredictability of the Internet behavior would pose very significant obstacles to the development of interactive distributed applications. At the present moment, the Internet is characterized by large delays on message transmission and by large variations in these delays. These characteristics are known as *latency* and *jitter*, respectively.

The difficulties in dealing with Multimedia data on the Internet are intensified when this data represents music. This happens because music presents much harder bandwidth and timing constraints than other kinds of applications. Currently, *Internet Music* is almost completely limited to one-way streaming of digitized audio with very little interaction. This paper discusses the viability of using the Internet for distributed musical applications addressing both artistic and technical concerns.

1.1 Musical Challenges

Music is a temporal art and each culture or each period in its history has managed to develop appropriate procedures to deal with timing matters in music. These procedures were determined by contextual constraints such as the development of musical instruments, the space where music was presented, the ability of performers in controlling the sound production with their instruments, and by the development of musical language itself.

For example, during Middle Ages, the acoustic resonance of large cathedrals where the Gregorian chant usually took place imposed some tempo constraints: a monophonic chain of long notes was necessary to guarantee the comprehension of the text that was sung. This situation was completely different during the Baroque and Classical periods when a more favorable acoustic, the technological development of musical instruments, and a sophisticated system of notations allowed a very refined manipulation of sound in relation to time.

In the current decade, the Internet is becoming a new space for the realization of music and, for this reason, it is necessary to develop new tools and procedures to handle the peculiarities of the Internet medium. These developments will only be possible with work by researchers in the music and computer science communities.

1.2 Technological Challenges

Multimedia data is not only voluminous but also highly dependent on timing constraints. When a server sends a 256kbps CD-quality audio stream through an overseas connection, it is essential that the network be capable of sustaining an average data rate of at least 256kbps. But this is far from being enough. If the audio is being played in real-time in the client side, it is also important that the network jitter be

small. In other words, the data must arrive at that rate all the time, it cannot stop for a fraction of a second and resume later.

The effects of network jitter can be minimized by buffering the data in the client side and delaying the start of the playback for some seconds. This is the technique adopted by all existing audio streaming software. Although it works well for some kinds of applications, it is intolerable for a large number of applications which requires a higher degree of interactivity.

Music performance is an extreme case of sensitivity to timing constraints during interaction. In applications such as video conference and collaborative work, a delay of up to one or two seconds is tolerable. However, in music, if two performers are playing together, a delay in the order of hundredths of a second is noticeable. In some cases, a delay of a tenth of a second might be fatal.

We have performed an informal experiment in the *Laboratório de Linguagens Sonoras* at PUC/SP using a MAX patch to delay the sound produced by one of the musicians playing a duet. We noticed that, after some practice, the musicians were able to compensate for delays of up to 70 or 80ms and still play together. Larger delays started to be intolerable.

The current Internet and Operating System infrastructures are not suitable for distributed music applications. However, a number of well-known computer science techniques developed in this decade can be used for making the dream of Internet Music come true. We believe that this dream not only can come true but that it will be true in the first decade of the next millennium.

Section 2 presents two examples of current work in Internet Music. Section 3 describes the technological problems that Internet Music faces and discusses possible solutions. Section 4 presents a variety of musical applications on the Internet. We present our conclusions in section 5.

2 Related Work

During the last decade, very little research has been carried out in the field of Internet Music. However, the recent explosion in the growth of the Internet and the fact that inexpensive personal computers are now capable of offering high-quality multimedia make this scenario likely to change.

NetSound [Casey and Smaragdis, 1996] is a sound and music description system that is capable of streaming *Csound* [Vercoe, 1997] specifications in real-time. The main advantage of this approach is an enormous economy in network bandwidth. Instead of sending a digitized audio wave through the network, *NetSound* sends a mathematical description of the sound wave which is then generated at the target machine. Therefore, instead of requiring bandwidth in the order of kbps or Mbps, it transmits data at rates in the order of hundreds of bits per second. The powerful *Csound* engine is used to synthesize high-quality audio in real-time at the target

machine.

Although this approach solves the bandwidth problem, it does not address latency and jitter issues (see section 3.3). Actually, it might increase these problems since intensive computation is performed in the target machine.

The *Global Visual-Music Project* [Puckette et al., 1998] is developing an infrastructure for real-time, networked multimedia for multi-site, improvisatory performance. This group has developed PD, a programming language for real-time audio and graphics applications. By using this language and a set of extensions called GEM [Danks, 1997], the group has composed *Lemma 1* for percussion and trombone. It was performed on September 1997 in a Jazz club during the International Computer Music Conference. This performance was the first phase of a series called *Global Visual-Music Jam Session.*

The second phase includes *Lemma Two* in which performers in two different locations will interact through networked digital equipment. Its first performance is scheduled for Fall 1998.

In order minimize quality of service problems with latency and throughput, the communication is done through dedicated ISDN connections. This approach makes its cost prohibitive for everyday use and for the vast majority of musicians and computer music researchers. A more generic solution is required.

These two research efforts bring important contributions to the future of Internet Music but there are still a large number of problems that must be addressed by future systems. We now describe each of these problems.

3 Technological Infrastructure

Most multimedia applications have in common the fact that they require real-time manipulation of large quantities of data. In the case of distributed multimedia, this data must be processed in, at least, two different machines and transmitted over a network.

Users dealing with applications such as video-on-demand and multimedia presentation systems can tolerate delays in the order of some seconds. There is no problem on waiting five seconds to watch a two-hour movie. When it comes to audio or video-conferencing tools, a delay of up to one second can be acceptable. If the delay is higher than that, the conversation will not flow smoothly.

Distributed multimedia system developers have experienced that existing operating system and networking infrastructures are not tailored to support the real-time guarantees that these applications require. The protocols on which the Internet is based were designed for exchanging textual information and for transferring files. They are not optimized for large amounts of data, for meeting deadlines, and for dealing with heterogeneous environments.

As mentioned before, Internet Music applications are the multimedia applications

that are most sensitive to timing constraints. In this section, we discuss how multimedia research has been trying to solve these technological problems and how their approaches can be applied to the challenge of making Internet Music a reality.

3.1 Data Compression

The first problem one faces when dealing with professional-quality music in a digital environment is the fact that audio data is voluminous. Thus, one can expect the need for large storage areas, high-bandwidth connections, and high processing power. Luckily, this is not only a problem for music. In general, video applications require much more resources than audio.

CD-quality audio requires two 16-bit tracks containing 44.1 thousand samples per second, which is equivalent to a data rate of 1411.2kbps. Recent research suggests that a higher sample rate and a larger sample size could still improve the human-perceptible sound quality. But this data rate already leads to a storage requirement of approximately 635Mbytes per hour.

This problem can be solved by applying data compression techniques. However, traditional compression mechanisms such as the Lempel-Ziv algorithm – implemented by programs like `gzip` – generally does not achieve a good compression rate when applied to audio data.

Audio-specific compression algorithms have been developed as a means to utilize the knowledge about the characteristics of the audio data in order to achieve a better compression ratio. The GSM algorithm [Vary and et al, 1988], for example, is used in audioconferencing and telephony applications and produces a data rate of 11.3kbps. But its quality is far from being enough for professional musical applications.

The MPEG layer 3 standard [MPEG.ORG, 1998] uses a psychoacoustic model [Sporer et al., 1992] to capture the information that is more relevant to the human ear and ignores what cannot be perceived. By applying filter banks, quantizations, entropy compression, and by exploiting redundancy of both channels, MPEG is capable of encoding high-quality audio at data rates ranging from 8kbps to 128kbps[1]. Near-CD quality can be achieved at 128kbps, resulting in a compression ratio of more than ten times.

The main problems that compression of high-resolution audio brings are the requirement for extra computational power and the delay that it imposes. Theoretically, MPEG layer 3 adds a delay of, at least, 59ms. The actual values, however, tend to be much higher than that depending on the implementation. Typically, existing hardware implementations add a delay of 150ms. Real-time encoding in software is not yet possible.

[1]The higher the data rate is, the higher is the sound quality.

3.2 Alternative Data Representations

As we saw, data compression adds a significant delay and is not always possible in software. Besides, high-quality compressed audio still yields large amounts of data and data rates in the order of 128kbps. These problems can be avoided by using alternative data representations rather than dealing with the digitized wave form.

The most commonly known alternative representation is the MIDI format that stores information about musical notes rather than sound waves [Loy, 1985]. The MIDI standard produces very low data rates typically in the order of 0.1 to 5kbps. The codified notes are received by the MIDI player which produces the output sound wave. This can be done either by manipulating wave tables or by synthesizing the wave forms from mathematical models. This process can be done both in software and using special hardware such as sound cards or other sound devices.

MIDI presents a series of limitations [Moore, 1987, Puckette, 1994] including the fact that the data stream does not contain a description of the wave form but, simply, pitch, intensity, and the "instrument" that plays each note. Each MIDI player is free to give its own interpretation of how the instrument will sound.

A better solution would be to extend the representation to include detailed information about the instrument's timbre. This approach is taken by the MOD format, first used in Amiga computers, which includes samples of the instruments' wave forms. However, MOD does not provide mechanisms for fine grain manipulation of the wave forms once they are transmitted. Total control of the sound wave is achieved in NetSound (see section 2) which transmits Csound specifications that are synthesized in real-time in the target machine.

We can clearly notice a significant trade-off in all these approaches. The higher is the control over timbre and wave form, the higher is the computation power requirement and processing delays. MIDI requires very little resources, incurs very little delays but does not provide control over the wave form. NetSound requires high-performance CPUs, imposes higher delays but enables the tightest control.

3.3 Networking Support

Research in networking support for real-time applications has been trying to solve three major problems: bandwidth, latency, and jitter. The **bandwidth** problem is the one that has received most attention from the industry and government. The bandwidth of existing wide-area networks has been increasing very rapidly in the last few years. According to Richard Palmer, a marketing director at Cisco Systems, Inc., new gigabit switch routers will scale from 622Mbps to 2.4Gbps in 1998 and to 9.6Gbps in 1999. *Internet2* [University Corporation for Advanced Internet Development, 1998] will connect more than one hundred research universities in the United States at 2.4Gbps by the year 2000. The same trend can be observed in other countries and in non-academic Internet systems.

These high-speed connections will enable the transmission of high-quality audio in real-time to a large number of sites. However, it does not solve the problems of latency and jitter.

In Internet Music, **latency** can be defined as the time between a musical event happens on one side of the Internet connection and the time in which it is reproduced on the other side. **Jitter** is the variance in the latency. Consider, for example, a MIDI keyboard that sends its output to a remote site over an Internet connection. On the other side of this connection, a synthesizer produces the sound corresponding to the notes sent through the connection. Assume a musician plays a C in the keyboard every second. The time elapsed from the instant in which the musician presses the C key and the instant in which the C is played on the other side is the latency l. But, l is not always the same. Depending upon the network and operating system state, hardware, and software, l can vary a lot from one note to the following one. This variance, which produces disastrous effects in musical applications, is called jitter.

Latency can be minimized and jitter can be almost completely eliminated by deploying new networking protocols and hardware. Commonly used Ethernet connections and the Internet Protocol does not provide any kind of guarantee regarding the transmission of data packets. One cannot know, a priori, what the available bandwidth is and the latency can vary from a few microseconds to several seconds.

ATM and FDDI networks [Steinmetz and Nahrstedt, 1995], on the other hand, provide the basis for resource reservation and admission control. On these networks it is possible to reserve a certain fraction of the network time to specific data streams. Therefore, the network provides the user some guarantees regarding bandwidth, latency, and jitter.

A new and much more powerful version of the Internet Protocol, called IPv6 [Deering and Hinden, 1997], provides the underlying support for implementing systems capable of streaming real-time data guaranteeing that bandwidth, latency, and jitter requirements are met. A large number of operating system vendors and research groups are currently implementing IPv6 on their systems. It promises to be the next standard for wide-area inter-networking.

Once IPv6 is combined with fast, real-time-capable networks such as ATM, it will be possible to rely on the network to deliver multimedia packets with a very low latency and jitter. We believe that in the first decade of the new millennium, personal computers will be able to connect distant musical spaces – within a country – with imperceptible latency and jitter. In fact, network quality of service is a key component of the research supported by the Next Generation Internet and Internet2 initiatives.

It is important to point out that our planet is large enough so that the network delay between certain points in its surface will always be perceptible. The distance between São Paulo and Tokyo – when traveling on the Earth surface – is 18,530km. Thus, traveling at the speed of light, a data packet will take no less than 61ms to go from one site to the other. This value is very close to the human perception limit

but current technology is very far from providing data transport with this kind of minimal delay. Therefore, it is likely that musicians playing together in these two locations will always perceive some delay between them.

3.4 Operating System Support

Existing commercial operating systems such as MS-Windows implement technology developed 25 years ago and, therefore, are suitable for meeting the requirements of applications used a quarter of a century ago. Multimedia and real-time applications have completely different requirements that have been addressed by recent research [Steinmetz and Nahrstedt, 1995].

Traditional operating systems manage hardware resources such as CPU, disk, memory, and network using best-effort strategies. They usually accept any number of tasks without paying attention to the quality of service that will be offered to their users. If one starts two video sessions and the compilation of a large program, the system will divide the CPU among all these tasks without taking into consideration the application-specific requirements. The video quality will probably be degraded even if there were enough resources for properly executing all tasks.

Real-time operating systems use special algorithms to support applications which must meet specific deadlines and receive a guaranteed portion of the machine resources. A video-on-demand application, for example, must be able to receive a certain amount of bytes per second from the network, receive a certain portion of the CPU time to decode the data stream and be able to send data out to be displayed on the screen and played on the speaker.

Experimental systems such as Nemesis [Leslie et al., 1996] controls the admission of new tasks and provide support for resource allocation and real-time scheduling based on application requirements. With these facilities, a system can offer guarantees that the multimedia streams will be processed with the required quality of service. Thus, the development of Internet Music applications are greatly facilitated on such environments.

4 Internet Music Applications

There are endless possibilities for applying Internet technology to music. In this section, we discuss some scenarios in which the topics discussed previously can be deployed.

4.1 Concert Broadcasts

At the University of Illinois, we have been developing a scalable distribution framework for real-time multimedia streaming [Kon et al., 1998]. Using this framework, we

ere able to build large distribution networks that could potentially serve 3,000 simultaneous clients with low-bandwidth video and audio.

Our technology was chosen to broadcast, live over the Internet, the coverage of the NASA JPL Mars Pathfinder mission [Golombek et al., 1997]. A network of more than 30 *Reflectors* spread across five continents was able to deliver live video and audio from the NASA Jet Propulsion Laboratory to more than one million clients in dozens of different countries.

This distribution framework is also capable of supporting the broadcast of CD-quality audio of a live concert or a MIDI-like stream. The only limitation is the available network bandwidth on the client Internet connections and on the Reflector sites.

Since this system currently runs on top of traditional operating systems and use the existing networking infrastructure, there are no guarantees in terms of bandwidth, delays, and jitter. We minimize this problem by buffering some data in the client before displaying it. This introduces additional delays that range from some tenths of a second to a couple of seconds.

Therefore, we have demonstrated that with existing technology it is possible to use the Internet for large-scale multimedia broadcasts and even for videoconferencing.

These delays are still too big for interactive music performances. But, since they are acceptable for videoconferencing applications, it would also be acceptable for master classes, workshops, and other forms of distant music learning.

4.2 Distributed Rehearsals and Concerts

Once the next generation of internet protocols are deployed and gigabit networks supporting quality of service are common, it will be possible to carry out distributed music performances. This will first be possible using dedicated links and, later, on the public Internet.

As happens with videoconferencing tools, each room participating in a distributed music session will be equipped with a conferencing system. This system will be responsible for (1) capturing the musical data generated on that room and sending it to the other participants, listeners, or to a Reflector, and (2) receiving musical data from remote sites and playing it locally.

Such a system will enable both distributed rehearsals and concerts. A conventional group, like a string quartet, would be able to "meet" several times a week to rehearse even if the group members live in different countries. The computer system will guarantee that the networking latency will not produce human-perceptible delays. High-quality audio and large-screen video will provide musicians with a rehearsing environment similar to the traditional one. If required, a limited number of "traditional rehearsals" could take place in the days preceding the concert.

However, even the concert could take the form of a distributed event with groups of musicians and audience, all in different locations. In that way, it will become much

easier for musicians from different places to interact with each other and exchange musical experiences and knowledge.

4.3 Remote Supercomputing Resources

Supercomputing brings us the opportunity of experimenting with tomorrow's computing resources. A supercomputer today has the computational power of personal computers of five to ten years in the future. Musical applications of supercomputing include real-time synthesis [Kriese and Tipei, 1992], and computationally intensive algorithmic composition tools such as *MaxAnnealing* [Iazzetta and Kon, 1995].

Supercomputers are, by definition, scarce resources to which very few people have access. They are usually shared by a large number of researchers and they have very little mobility. Therefore, it is very unlikely that a supercomputer could be taken to a concert hall in order to perform music in real-time.

However, proper network support would enable the real-time connection of a supercomputer to other computers in a concert hall. Musicians could interact with local devices that would forward information to remote locations where the supercomputing power is available. Upon completion of the computation (e.g., sound synthesis, or algorithmic composition of parts of a piece) the result would return to the concert location and local systems would turn the data into music.

4.4 Composition of Internet Music

While final solutions for the technological problems are not found, we can think of a number of issues that contemporary composers should keep in mind when composing music for the Internet.

As stated in section 1.1, the design of interactive music systems on the Internet must take into account the peculiarities of the specific types of music they are intended for. What may be seen as extremely relevant in a particular context may not be pertinent to a different musical situation. There are at least three timing factors that will guide the design of an Internet music system.

1. the existence of a regular pulse: pieces based on regular timing structures (pulse, beats, bars) are much more sensitive to delays and jitters. Certain kinds of improvisational performances would be less affected by these constraints;

2. tempo: the tolerance on timing deviations is inversely proportional to how fast a piece is played and to the rate at which the events occur. Thus, a 100ms delay can be irrelevant for a slow piece but very disturbing for a very fast one.

3. compositional style: in a free improvisation – as the ones found in some of George Lewis and John Cage's pieces, for example – the performer can build musical structures as he receives the sound data from the Internet. On the other hand, in pre-composed music, musicians on both sides of an Internet connection must be listening to the same information at the same time, or with a very short delay.

It is known that a trained musician is able to compensate for small timing deviations, as in the case of a timpanist who plays slightly ahead of the string session in a symphonic orchestra to compensate the distance the sound must travel from the back of the stage to the audience. But the delay must be very small and constant. Since there will always be a delay in the transmission of information over the Internet the issue that must be addressed here is the following. For a specific kind of music, how small the delay must be to provide a satisfactory level of interaction between different players.

By keeping in mind the specific characteristics of this medium, an Internet Music Composer can compensate for the current technological limitations. At the current stage of the Internet development, having a good knowledge about existing technical limitations is fundamental for the composition of Internet Music. As technology advances and the problems described in this paper are addressed are solved, it will become increasingly easier for composers to utilize this new medium.

5 Conclusions

Within the computer music community there is an increasing interest in the Internet in general and in Internet Music in particular. In spite of that, very few research groups and composers are working with distributed computational systems for music performance, composition, or education.

We have demonstrated that, although the existing computational infrastructure still presents significant problems, they all can be solved. Recent research in the areas of real-time systems, quality of service for multimedia, networking, and distribution mechanisms assure that a sound set of mechanisms can be combined to produce an ideal environment for distributed music applications on the Internet.

Composers can expect to have access to these environments in the first decade of the next millennium. But, one must not wait for them passively as there are important aesthetic questions regarding the use of the Internet for music practice. These questions are to be posed by the artistic community and answered by research in art, science, and technology.

References

[Casey and Smaragdis, 1996] Casey, M. and Smaragdis, P. (1996). NetSound. In *Proceedings of the 1996 International Computer Music Conference*, Hong Kong.

[Chen et al., 1996] Chen, Z., Tan, S. M., Sane, A., Li, Y., Campbell, R., and Xie, D. (1996). Video and Audio: Organization and Retrieval in the WWW. White Paper. Available at `http://vosaic.com/corp/papers/www5.html`.

[Danks, 1997] Danks, M. (1997). Real-time Image and Video Processing in GEM. In *Proceedings of the 1997 International Computer Music Conference*, pages 220–223, Thessaloniki.

[Deering and Hinden, 1997] Deering, S. and Hinden, R. (1997). *Internet Protocol, Version 6 (IPv6) Specification.* draft-ietf-ipngwg-ipng-spec-v2-01.txt. Internet Draft.

[Golombek et al., 1997] Golombek, M. P., Cook, R. A., Economou, T., Folkner, W. M., Haldemann, A. F., Kallemeyn, P. H., Knudsen, J. M., Manning, R. M., Moore, H. J., Parker, T. J., Rieder, R., Schofield, J. T., Smith, P. H., and Vaughan, R. M. (1997). Overview of the Mars Pathfinder Mission and Assessment of Landing Site Predictions. *Science*, 278(5344):1734–42.

[Iazzetta and Kon, 1995] Iazzetta, F. and Kon, F. (1995). MaxAnnealing: A Tool for Algorithmic Composition Based on Simulated Annealing. In *Proceedings of the Second Brazilian Symposium on Computer Music.*

[Kon et al., 1998] Kon, F., Campbell, R. H., Tan, S., Valdez, M., Chen, Z., and Wong, J. (1998). A Component-Based Architecture for Scalable Distributed Multimedia. In *Proceedings of the 14th International Conference on Advanced Science and Technology (ICAST'98)*, pages 121–135, Lucent Technologies, Naperville.

[Kriese and Tipei, 1992] Kriese, C. and Tipei, S. (1992). A Compositional Approach to Additive Synthesis on Supercomputers. In *Proceedings of the 1992 International Computer Music Conference*, pages 394–395, San Jose, CA.

[Leslie et al., 1996] Leslie, I., McAuley, D., Black, R., Roscoe, T., Barham, P., Evers, D., Fairbairns, R., and Hyden, E. (1996). The Design and Implementation of an Operating System to Support Distributed Multimedia Applications. *IEEE Journal on Selected Areas in Communication*, 14(7):1280–1297.

[Loy, 1985] Loy, C. (1985). Musicians Make a Standard: The MIDI Phenomenon. *Computer Music Journal*, 9(4):9–26.

[Moore, 1987] Moore, F. R. (1987). The Dysfunctions of MIDI. In *Proceedings of the 1987 International Computer Music Conference*, pages 256–263, San Francisco.

[MPEG.ORG, 1998] MPEG.ORG (1998). *MPEG-Audio layer 3 resources.* URL: `http://www.mpeg.org/index.html/mp3.html`.

[Puckette, 1994] Puckette, M. (1994). Is there life after midi? In *Proceedings of the 1994 International Computer Music Conference*, page 2, Aarhus, Denmark.

[Puckette et al., 1998] Puckette, M., Sorensen, V., and Steiger, R. (1998). The Global Visual Music Project. Project Home Page: `http://www.earunit.org/gvm.htm`.

[Sporer et al., 1992] Sporer, T., Brandenburg, K., and Edler, B. (1992). The use of multirate filter banks for coding of high quality digital audio. In *Proceedings of the 6th European Signal Processing Conference*, pages 211–214.

[Steinmetz and Nahrstedt, 1995] Steinmetz, R. and Nahrstedt, K. (1995). *Multimedia: Computing, Communications, and Applications.* Prentice Hall, New Jersey.

[Tan et al., 1996] Tan, S., Campbell, R., Chen, Z., Liao, W., Raila, D. K., Kon, F., and Valdez, M. (1996). Adaptation and Synchronization in Low-Bandwidth Internet Video. In *World Wide Web Consortium Workshop on Real Time Multimedia and the Web (RTMW '96)*, INRIA Sophia Antipolis, France.

[University Corporation for Advanced Internet Development, 1998] University Corporation for Advanced Internet Development (1998). Internet2 Project Home Page. `http://www.internet2.edu`.

[Vary and et al, 1988] Vary, P. and et al (1988). A speech codec for the European mobile radio system. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 227–230.

[Vercoe, 1997] Vercoe, B. (1997). *Csound Manual.* Media Lab, MIT. Edited by Jean Piché.

Acknowledgments

The authors gratefully acknowledge the support provided by CAPES and FAPESP. Fabio Kon is supported by a grant from CAPES, proc.# 1405/95-2 and Fernando Iazzetta is supported by FAPESP, proc.# 97/01688-1.

From NIFF to Musical Braille

Didier Langolff, Pierre Gradit, Nadine Jessel, Monique Truquet (TOBIA)
Institut de Recherche en Informatique de Toulouse
Université Paul Sabatier
E-mail : langolff@irit.fr

1. Introduction

Our purpose is to make faster the transcription of printed score into Braille. After a study of several Musical notation "standards", we decided to develop a software which will make the transcription from NIFF (Notation Interchange File Format) to Musical Braille. In the following paragraph we will present some key concepts of the NIFF notation, NIFF standard is a computer-oriented description of printed scores. Then we will describe the transcription software called "Editor" and some specific features.

The two used languages (NIFF and Musical Braille), are probably near as their main constraint is the same: the writing of music scores being made with a single sequence of characters. The basic step of our method was to find a way of editing musical Braille near the description of a score in a NIFF notation. So, the first step was to structure the Braille characters sequence with a tree-structure to develop then a complete editor of this structure.

2. NIFF notation

NIFF is a file format designed to allow the interchange of music notation data between and among music notation editing and publishing programs and music scanning programs. Its design is a result of combined input from many commercial music software developers, music publishers, and experienced music software users[1] .

The lack of an accepted standard format for music notation has been during several years a source of great frustration for computer musicians, engravers and publishers. Numerous attempts have been made in the past to create a standard format. The effort resulting in NIFF is interesting for us for different reasons:

- NIFF files can be extracted from printed scores with different processes. We opted for the scanning program MIDISCAN which proposes a NIFF automatic

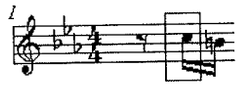
[1] NIFF 6.A, Notation Interchange File Format, 1995. http://mistral.ere.umontreal.ca /~belkina/NIFF.doc.html

translation, the other way being the result (the NIFF output) of editing programs. One of the main feature of NIFF is that the description of a score in NIFF format is purely graphic. For example, a note is described not by its value (A, B, C...) but by its position on the staff.

NIFF basic structure is a tree (See Appendix 1). A NIFF file is divided into two sections: the Setup Section, containing general information relative to the whole score, and the Data Section, containing the music symbols and layout information. The tree structure is inherent of the NIFF file structure, but also of the use of the anchor concept. We will describe in the following paragraphs these key features of NIFF format.

2.1 Lists and "chunks"

Lists and "chunks" are the basic components of a NIFF file. All the lists and "chunks" available to create a NIFF file are presented in the NIFF specifications[1].

Notehead "chunk" specification		**Printed score sample**	**Interpreted NIFF code**
Notehead	**Chunk**		
Shape	**Byte** 1 = double whole 2 = whole 3 = half-whole 4 = quarter-whole ... 14 = open triangle		note: [6] (4 , 5*, 0 , 1 , 0, 16**)
Staff step	**SIGNEDBYTE**		
Duration	**RATIONAL**		
Default anchor:	Time-slice		
Default reference point:	The center of the bounding box of the notehead		"[6]" is the size of the body. *: Position on the staff, first line has the value 0. **: Duration of the note: 1/16th
Tags:	Part ID, Voice ID, placement tags, MIDI, Performance, Grace Note, Cue Note...		

2.1.1 Data "chunk"

The "chunk" may be separated in two classes: setup "chunk" and symbol "chunk" .Symbol "chunks" describe the music score (See example above). Setup "chunks" define the context of the description.

A "chunk" is the basic structure of a NIFF file (which is an extension of RIFF (Resource Interchange File Format)). A "chunk" is composed of a header and a body. The header is written over 8 bytes. Four bytes give the name of the "chunk", and four give the size of the body.

In the following table we present a NIFF file sample concerning the notehead "chunk".

2.1.2 Lists

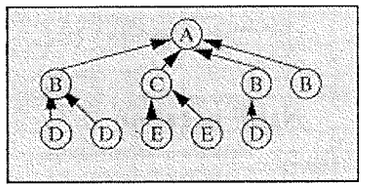
A list is a type of "chunk" which can contain nested "chunks". Its code of four characters is "LIST". The type of the list consists of a "four characters code" following the size field, which is followed by a series of chunks or other lists. Pages, systems, and staves are all represented by lists.

Staff "list" specification		**NIFF notation**
Staff	**LIST**	LIST:[2382]syst
		syhd:[0]()
Location:	System list	
Required:	Staff Header chunk	LIST[1]:[1252][2] staf[3]
Optional:	Any number of the following:	Sthd:[18](...)
	Time-Slice chunk	...[4]
	Any music symbol chunk	
	NIFF Font Symbol chunk	
	Custom Shape Graphic chunk	1: Four characters code (always "LIST")
	Text chunk	2: Size of field
	Line chunk	3: Type of list
		4: series of "chunks" or other lists

The "chunks" given in the body of a list "chunk" are the child of this "chunk". List "chunks" generate a tree, called reading tree, where nodes are the list "chunks", and leaves the data "chunks". The last level of the reading tree is the staff. So all symbols describing a staff are at the same level as the reading tree. But, informally, the tree structure continues deeper than the staff level. Measure, Stem, Notehead could be considered as nodes of a tree, taking into account the concept of anchor/dependant relationship.

2.1.3 The anchor/dependant relationship

In NIFF, a music symbol whose placement depends on one or more other symbols is called a "dependent" symbol, and the symbol or other "chunk" type on which its placement depends is called its "anchor." For each symbol "chunk" type, a default anchor "chunk" type is defined (except for the root: the page header "chunk"). An example of this is given in the above table concerning the representation of the notehead "chunk": you can see that the default anchor of the notehead is the "time-slice". This means that every symbol "chunk" is child of the last previous "chunk" corresponding to the anchor type. For example, let be (A ← B, A ← C, B ← D, C ← E) a set of (Anchor ← Dependant) relationship. Then to the following sequence *ABDDCEEBDB*, corresponds the following tree structure:



Where:

A: could represent a time-slice (measure-start event),

B: could represent a notehead "chunk",

C: could represent a stem "chunk",

D: could represent an accidental "chunk",

E: could represent a chord "chunk".

2.2 Interest for our "Editor software"

So, there are two "tree structures" in a NIFF file:

- Explicit tree: the reading tree, which nodes are lists and data "chunks" are leaves. This tree is the NIFF file.
- Implicit tree: The final tree, in which data "chunks" may be nodes. The basic notion associated to the final tree is the anchor concept and the memory table. To build this tree, external information is needed, common to all NIFF files: the anchor/dependant relationship.

In this software, we give the notes relatively to their height in music (A,B,C...), their octave is given only if needed. We adopt this notation (different from NIFF notation) only to correspond to the Musical Braille definition.

3 The Software

3.1 Musical Braille notation for stave description

We now focus on musical Braille notation. The first official normalization of this notation takes place in the nineteenth century (Cologne Key 1888), of course without any global structure description as a regular grammar. For the transcription, we choose to use the musical Braille notation described in the New International Manual of Braille Music Notation[2].

But as a staff depicted in Braille is a sequence of characters, other musical objects are in fact represented by "sub-sequence". These sequences, which represent a musical object, are called frames. Frames studied in our work are regular, that is to say if a child frame starts in a father frame, it has to end before this father frame.

These structures may be represented by trees. In those trees, called frame trees, nodes are frames and leaves are the characters.

A Braille character has 6 dots[3] which permit to obtain only 64 combinations. So in some cases we need several Braille characters to represent only one classical item. For example the "key" notation requires 3 Braille characters. On the contrary, a single Braille character represents a note or a rest (name and value). The names of the notes are obtained with the dots 1,2,4 and 5; the dots 3 and 6 determine the note values.

3.2 Tree concept

It is now clear that the tree structure is the main structure used in our work. As we developed our software in an object-oriented language, we defined abstract class defining trees.

The software may be seen as trees manipulation. Many instances of the tree structure are used:

- the frame tree: the internal describing language of the software, which nodes are frames and leaves are characters. The action associated to score, page and line is *void update()*, that guarantees the validation of pagination constraints (number of characters per line, hyphenation, number of lines per page). Updating is automatic.

---

[2] New International Manual of Braille Musical Notation, Bettye Krolick, World Blind Union, 1996.

[3] 3 lines, 2 columns.

the score tree: the Braille display format, which nodes are score, page, line and the leaves are characters.

the reading tree: the NIFF file format, the element action is *int read()*, *read* returns the number of bytes read. This is an automatic action.

the intermediate tree: the intermediate format, the element action is *void translate()*. It gives the corresponding frame and inserts it in the context (contextual insertion).

the dictionary tree: the database organization using the standard file system, which nodes are folders and leaves are files. The element action is *void insert()*. It inserts in the context the frame described in the selected file.

3.3 Editor: set of basic tools

We now focus on tools, components of the editor (See Appendix 2). All these components are browsers of tree.

The Cursor viewer is a frame tree browser.

The Braille panel is a score tree browser.

The NIFF translator is an intermediate tree browser.

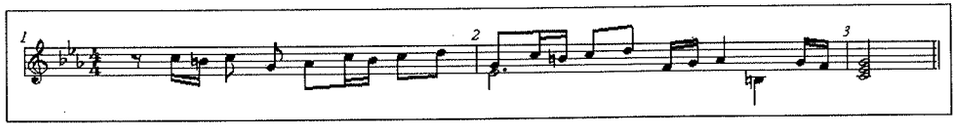
The Dictionary is a file tree browser.

Each browser has its own cursor. So at a time, only one browser and only one element of the corresponding tree are active. The state display of the software is the display of the String representation of the current element. The set of command at this instant is:

- Change active browser (Next Tool/ Prev Tool).
- Move in the current tree (Next/Prev/Outside/Inside)
- Action the current element (Action)

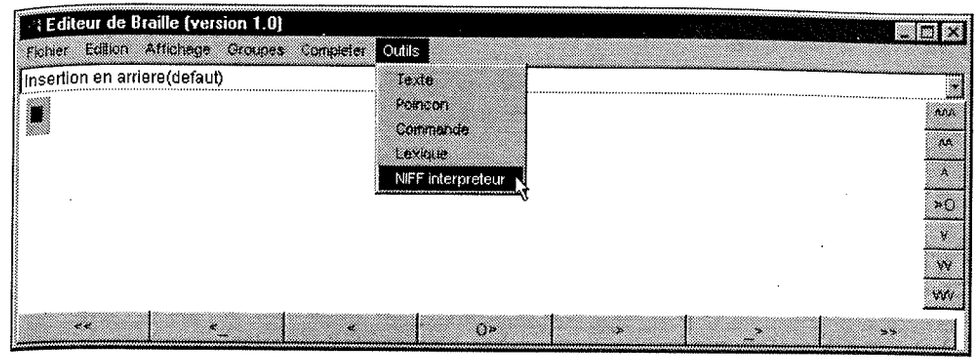
3.4 Editor: Example of use

For a better understanding, we will develop an example of transcription with the following staff:

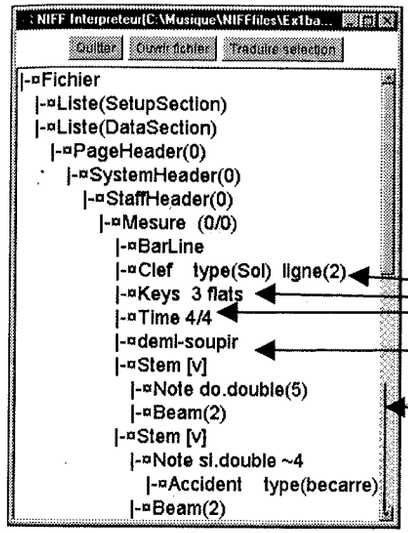


The first step is to acquire a NIFF file of this staff, for example with MIDISCAN (as mentioned in the paragraph 2).

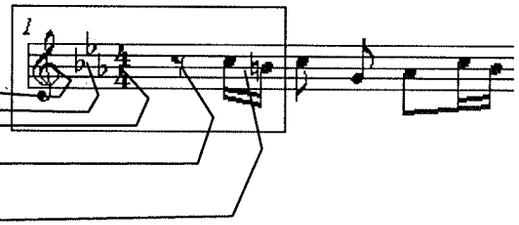
We set up the program and we chose to begin the transcription, so we have to activate the "NIFF interpreteur".



This program takes the NIFF file corresponding to the above staff, and gives the following tree.



This section corresponds to the framed part of the score:

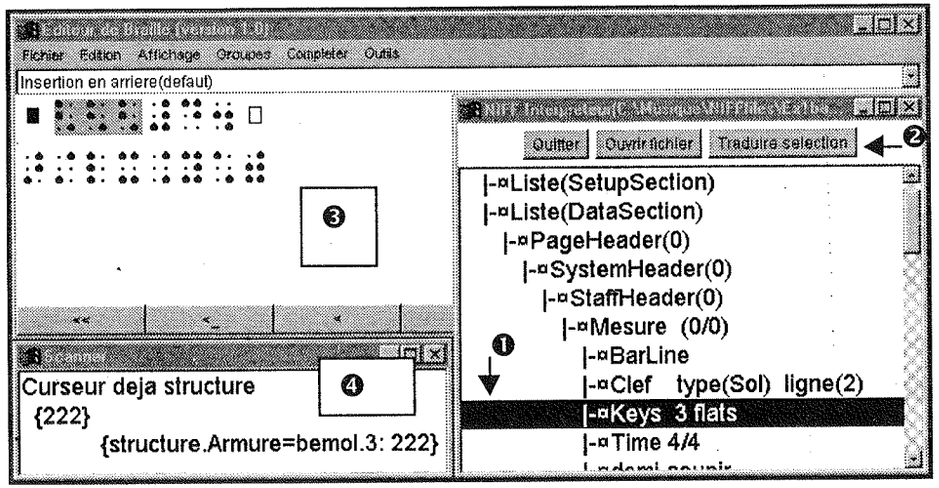


✵: Intermediate tree

At the beginning, you have only the root of the tree (called "Fichier") in the windows of the NIFF translator.

You can expand the tree by double clicks on the nodes until a leave is encountered. For example, when you activate the node fichier, the nodes Liste(SetupSection) and Liste(DataSection) appear, according to the NIFF structure (See Appendix 1).

When you are in the NIFF Translator, you can browse the file (Intermediate tree) and decide to transcribe what you want by selecting the part (❶)you have to activate the transcription button (❷):



In the above example, the user selects the key signature and actives the transcription button. The result of the operation appears in two different windows if they have been opened by the user:

- The Braille panel (❸) where we can see the current translation highlighted.
- The window called "Scanner" (❹) which permits to browse the frame tree.

In the transcription step, the user can translate the whole score, a part of a score or just one item of a score.

4 Some specific features

4.1 Contextual insertion

The contextual insertion manages automatically the insertion of symbols into frames at the right place. The Braille notation enforces some rules in the order of the appearance of musical items. For example, if the Editor has to transcribe a sequence containing a note, an

accident and a dot, according to the Braille notation, the definitive sequence will be: the accident, the note and the dot.

A basic attribute called priority is used to manage contextual insertion. The routine *int getPriority()* returns a value, which permits to insert each frame at the right place (according to the Braille notation).

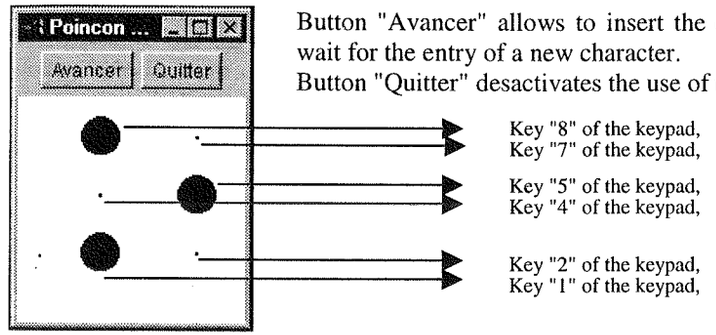
4.2 Creation of Braille Scores

The user has the possibility to create a score directly in Braille notation (without a NIFF translation). The software offers two different possibilities:

- The user knows the Braille musical notation and use indifferently the tools called "poinçon" or "dictionary".
- The user doesn't know the Braille musical notation and he has to use only the "dictionary".

4.2.1 The "Poinçon"

The "poinçon" is a tool available in the "tools menu" of the Braille panel. The user can write Braille characters via the keypad and the keys 1, 2, 4, 5, 7, 8.

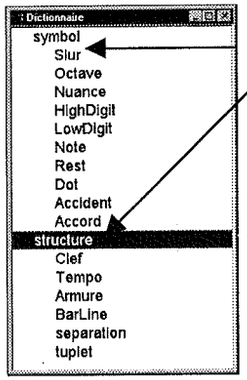


Button "Avancer" allows to insert the current character and wait for the entry of a new character.
Button "Quitter" desactivates the use of the "poinçon"

If an user does not find a Braille character or a sequence of Braille characters in the dictionary he can create them and store them in the dictionary with the help of the "poinçon".

4.2.2 The "Dictionary"

Based on the Dictionary tree (Cf. paragraph 3.2), the dictionary tool allows to an user (knowing or not Braille notation) to write a score according to the Braille notation. The principle is the following: the user has to browse the Dictionary tree, choose the element he wants to insert, and a double click allows to insert the element at the right place.



The dictionary tree has two main nodes:
Symbol and
Structure.
You have to develop the nodes as it's made in the example (on the left), and to go on with the other nodes until the needed leave is encountered.
The leaves of the Dictionary tree correspond to the Braille characters.

5 Conclusion

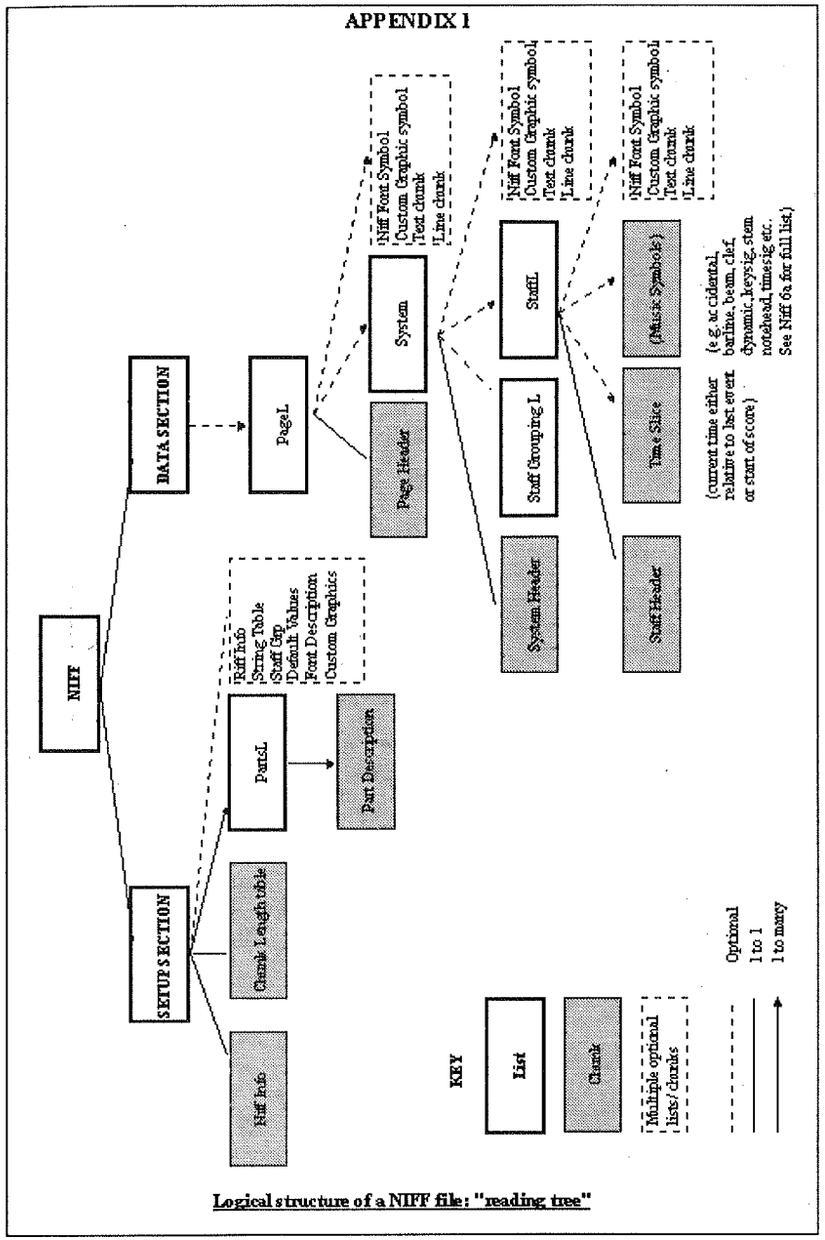
At the present time, the software is accessible by a sighted person. This person, who could know or not the music Braille notation, can transcribe a part of a score. Some parts of the Braille transcription software have to be developed in order to transcribe the whole score.

We plan to adapt this software to allow a blind person to use it with the help of a screen reader and text to speech synthesis. The use seems to be possible due to some features of the software:

- Only one browser is active at one time.
- The four keypad arrows represent the moves (left, right, outside, inside).
- The space key triggers the action associated to the element.
- Page-Up/Page-Down change the current tool.
- Some keys of the Braille sensitive line could be used too.

Another extension of this software could be used for the learning of music. In the example presented above, we saw that it's possible to browse a score and to know at any

time the meaning of each symbol, in Braille notation and classical notation as well (these indications appears in two separate windows). With the help of multimedia devices and the improvement of our software, it would be possible to listen to the whole score or a part of the score.



Logical structure of a NIFF file: "reading tree"

Designing a Sound Object Library

Victor Lazzarini e Fernando Accorsi

Núcleo de Música Contemporânea
Departamento de Arte / Departamento de Ciência da Computação
Universidade Estadual de Londrina

Abstract

The Sound Object Library is being designed to be employed by programmers and composers to develop sound manipulation applications. Its classes encapsulate all the processes involved in sound creation, manipulation and storage. The library can be used as a framework or, alternatively, as a set of objects to be patched together in a program. It is being developed to be portable across the UNIX and Windows platforms. The library class hierarchy is founded on three base classes, corresponding to sound processing objects, maths function-tables and sound input/output objects. A number of classes have already been implemented. A programming example shows one of the possible applications of the library objects. A GUI interface for the objects is proposed, using the V C++ framework.

1 Introduction

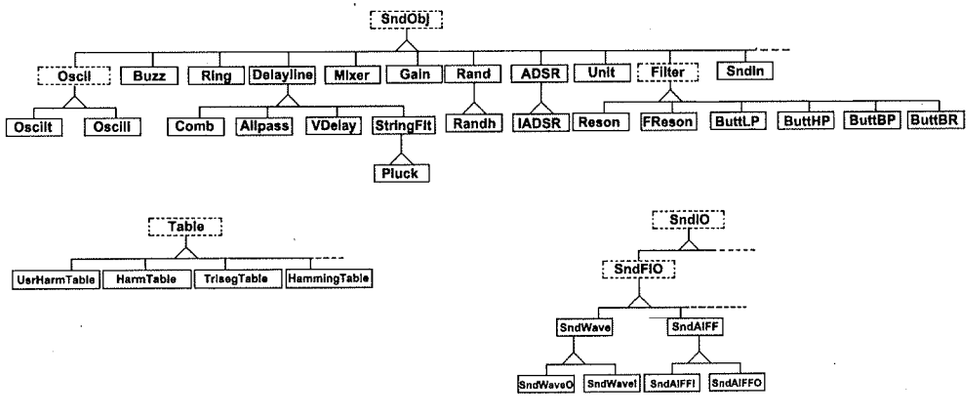
The research described here derived from some ideas that evolved during the work on the software *Audio Workshop* (Lazzarini, 1998). In that process, it was noted that the development of a set of objects for audio signal processing could be very useful in the design of new programs. It would help provide higher level tools for audio programming and software with a more intuitive user interface. This set would be used in two ways: as finished objects, to be employed directly in a program (or in a visual patching application), and as a framework, to which developers could add their own specialized code. Consequently, programs to carry out specific tasks would be easily developed by connecting the available objects and providing standard control-of-flow. Also, a text-based sound compiler could be designed to make use of the library. Using derivation and inheritance, new objects could be created from the existing ones, thus leaving the development possibilities open. The creation of a Sound Object Library seemed a worthwhile, though complex, task to pursue.

The project took shape as some important premises were being defined. Firstly, the library objects should encapsulate all the processes involved with production, manipulation and storage

of audio data. Secondly, the user would have to be able to freely associate the objects, as if they were modules in an analog synthesizer, or opcodes in systems such as, Csound (Vercoe, 1992) and Cmusic (Moore, 1990). Finally, the core of the code should be portable, allowing for machine-dependent specialisations when necessary. This, together with other practical reasons, lead to the choice of a portable high-level language such as, C++ (Stroustroup, 1991), with compilers available for a great number of machines. The project has been implemented, initially, on two UNIX platforms, Sun SparcServer and ULTRASparc under Solaris, and IBM RISC 2000 under AIX, as well as on Pentium PCs under Windows 95 and Windows NT. Because of its availability across all the target platforms, including the Windows operating systems, the chosen development environment was that of the Gnu Project, including g++, gdb and the Emacs editor. In the Windows platforms, MS Visual C++ was also used, since it compiles to an object code different to that of the Cygnus Windows Gnu Project (cygwin) c++ compiler. The programs created with the cygwin tools also require a special DLL to run, although their performance seems to be acceptable. An alternative port of the gnu compiler for the win32 operating system was also used. It is provided by the Minimalist Gnu Win32 (Mingwin32) tools, which do not require a runtime DLL and therefore have a better performance.

2 The Sound Object Library hierarchy

The proposed hierarchy for the library is based on three abstract base classes: SndObj, Table and SndIO. These form the base for three types of objects that integrate the set, respectively: sound processing, maths function-table and sound input/output objects. A diagram showing the inheritance relationships (Rumbaugh *et al*, 1994) between the classes in the library is shown below:



The classes derived from SndObj are involved in the production and manipulation of sound samples and they can make use of the Table classes. The SndIO tree is dedicated to all the processes involving sound input or output: disk, ADC/DAC, screen printing, etc.. These classes use a SndObj as their input and a SndIn class (derived from SndObj) can receive a SndIO-derived input, completing the link with the real world. In order to provide sound buffering services, a circular buffer class, SndBuff, was implemented and is used by the SndIO-derived classes.

2.1 The SndObj classes

A brief description of the interface for the SndObj base class is shown below:

```
class SndObj {
        protected:
      float* output; // output samples buffer
      float  m_sr;      // sampling rate
      int    m_error;   // run-time error code
      short  m_enable;  // object status (default ENABLED)

        public:
      void   Enable();  // enable object
      void   Disable(); // disable object
      float  GetSr();   // get the sampling rate
      void   SetSr(float sr); // set the sampling rate
      float  Output();  // get the current output sample
      virtual char* ErrorMessage(); // error messages
      virtual short DoProcess();    // processing
            };
```

The sound objects have one output, a sampling rate, an on/off switch and an error code. They also have a `DoProcess()` function that carries out all the due processing of a particular object and other methods to access its member variables. The derived classes have an undefined number of inputs (objects and/or set values) and some of them also rely on maths function-table objects to do their processing having one or more members of the Table type. Some examples of SndObj classes already implemented are: Oscilt and Oscili, two types of oscillators; Mixer, a sound object mixer; Gain, a gain modifier; Rand and Randh, two types of noise generators. Many other classes, such as envelope shapers, filters and delay-line objects were also implemented and some spectral-domain processing objects are being developed. The sound processing objects are designed to be easily interconnected, as it will be shown in the programming example. Most of them can receive one or more SndObj inputs. For instance, Oscillators can receive any sound object as their

frequency and amplitude inputs. Filters receive SndObjs as their audio input, as well as their frequency and bandwidth inputs. Mixers can receive any number of input objects and mix them together. The ability to make patches of processing boxes gives the flexibility necessary for users to create a great number of applications. As an example of a SndObj-derived class, an abbreviated definition of the Mixer class is shown below:

```
class Mixer : public SndObj{
        protected:
      SndObjList*  m_InObj;  // list of input SndObj
      int          m_ObjNo;  // number of input objects

        public:
      Mixer();   // constructors and destructor
      Mixer(int ObjNo, SndObj** InObjs);
      ~Mixer();

      int GetObjNo(); // return number of inputs
          short AddObj(SndObj* InObj); // add an SndObj to the
                                       // input list
      virtual short DoProcess();   // mixing operation
      virtual char* ErrorMessage(); // error messages
                        };
```

This object can receive any number of SndObjs as its inputs, adding them to an input list. The mixing is done by summing the output samples of the input objects in the `DoProcess()` method. An instance of the Mixer class can be created either by passing the number of initial SndObj inputs and an array of SndObj pointers or as an empty object whose inputs are going to be set later by the `AddObj(...)` method.

2.2 Tables

The table classes were developed to supply certain sound objects with tabulated maths functions. A common use for them is to provide one cycle of a certain waveshape to be continuously sampled by an oscillator. The HarmTable is an example of such an object that creates any of the following four harmonic waveforms: sine, saw, square or pulse (buzz). It can be used by an oscillator to create a pitched sound of a particular timbre. Similar to that, is the UsrHarmTable, which allows the user to define the relative amplitude of the individual harmonics. Another common type of function table is to store a shape to be used as an amplitude or frequency envelope. The TrisegTable class creates a three-segment line, with the option of logarithmic or linear lines, that can be used by an oscillator to control a parameter of some other sound object. Also, a

generalized Hamming window function table is supplied, as the HammingTable object. Table objects are very useful and can be employed in a variety of ways and will be constantly added to the library implementation.

2.3 Input and Output

The SndIO classes are designed to deal with all the input and output services need by the sound objects. They provide the `Read()` and `Write()` methods that will be used to perform the IO functions, regardless of whether the target is a soundfile, the ADC/DAC, the computer screen or some other device. The derived classes are designed to fit into these main categories: soundfile IO, which will have derived classes for each format, DAC/ADC IO and screen output, both having derived classes that will be platform-dependent. The resumed description of the interface for the SndIO class is shown below:

```
class SndIO {
        protected:
      float* output;   // output sample buffer
      SndBuffer* m_SndBuff; // internal buffer
      float m_sr;          // sampling rate
      int m_error;         // run-time error code

        public:
      float GetSr();      // get the sampling rate
      float Output(int channel);// get the current output sample
                                // for the specified audio
                                // channel
      virtual short Read();     // read from device
      virtual short Write();      // write to device
      virtual char* ErrorMessage(); // error message
        };
```

Some classes of the input and output tree were initially implemented. SndFIO is a SndIO-derived abstract class that controls file I/O, from which other classes, dealing with specific soundfile formats, are derived, like SndWave and SndAiff. Two classes are derived from SndWave, SndWaveI and SndWaveO. They perform RIFF-Wave file input and output, respectively. SndAiffI and SndAiffO are SndAiff-derived classes that deal with AIFF soundfile format. A special sound object, SndIn, can receive one-channel sound input of SndIO-derived object. Its output can be used by any SndObj-derived object in the processing chain. For multichannel input, multiple SndIn objects can be used.

3 Programming example

The following application uses some of the Sound Object classes to simulate the ingenious Jean-Claude Risset design (Dodge & Jerse, 1985). The output sound is a cascading harmonics drone, created by the minute differences in frequency of the nine oscillators employed. This programme, named *Risset*, can be called with the following command line:

```
Risset filename.wav duration(s) fr(Hz) amp(dB) no_of_harmonics
```

This application consists basically of a `main()` function that uses the Sound Object Library classes. It uses four types of sound objects: `Oscilt`, a truncating oscillator; `Oscili`, an interpolating oscillator; `Gain`, a gain control; and `Mixer`, a sound object mixer. The code can be divided in two parts: the creation and patching of the objects and the synthesis loop. In the first, all the boxes are set and connected: the mixer receives all the nine interpolating oscillators as inputs, which in turn receive the truncating oscillator as their amplitude input. The `SndWaveO` sound output receives the gain object, which attenuates the mixer signal. The synthesis loop is simply based on the ordered calls to the `DoProcess()` methods of the respective objects, followed by a call to the soundfile output `Write()` member function. The complete code (with the exception of the `usage()` function) for the Risset program is shown below.

```
//*******************************************************//
//      RISSET.CPP                                       //
//      Sample application using Sound Object Classes    //
//      synthesizes a cascading harmonics drone, as      //
//      designed by J C Risset                           //
//                  Victor Lazzarini, 1997               //
//*******************************************************//
#include <iostream.h>
#include <stdlib.h>
#include "AudioDefs.h"
void usage();

int
main(int argc, char* argv[]){
    if(argc != 6){
    usage();  // usage message
    return 0;
             }

// command line arguments
float fr = (float)atof(argv[3]);          // frequency
```



```
float amp = (float)atof(argv[4]);         // amplitude
float duration = (float)atof(argv[2]);    // duration.

// Envelope breakpoints & function table object
float TSPoints[7] = {.0f, .05f, 1.f, .85f , .8f, .1f, .5f};
TrisegTable envtable(512, TSPoints, LINEAR);

// Wavetable object
HarmTable table1(1024, atoi(argv[5]), SQUARE);

// truncating oscillator object (envelope)
Oscilt envoscil(&envtable, 1/duration, 32767);

// 9 interpolating oscillator objects
Oscili oscil1(&table1, fr, 0.f, 0, &envoscil);
Oscili oscil2(&table1, fr-(fr*.03f/110), 0.f, 0, &envoscil);
Oscili oscil3(&table1, fr-(fr*.06f/110), 0.f, 0, &envoscil);
Oscili oscil4(&table1, fr-(fr*.09f/110), 0.f, 0, &envoscil);
Oscili oscil5(&table1, fr-(fr*.12f/110), 0.f, 0, &envoscil);
Oscili oscil6(&table1, fr+(fr*.03f/110), 0.f, 0, &envoscil);
Oscili oscil7(&table1, fr+(fr*.06f/110), 0.f, 0, &envoscil);
Oscili oscil8(&table1, fr+(fr*.09f/110), 0.f, 0, &envoscil);
Oscili oscil9(&table1, fr+(fr*.12f/110), 0.f, 0, &envoscil);

// Mixer & gain objects
Mixer  mix;
Gain   gain((amp-15.f), &mix);

//  Add the oscili objects to the mixer input
mix.AddObj(&oscil1);
mix.AddObj(&oscil2);
mix.AddObj(&oscil3);
mix.AddObj(&oscil4);
mix.AddObj(&oscil5);
mix.AddObj(&oscil6);
mix.AddObj(&oscil7);
mix.AddObj(&oscil8);
mix.AddObj(&oscil9);

// output to a RIFF-Wave file
SndWaveO  output(argv[1], 1, 16, OVERWRITE);
output.SetOutput(1, &gain);

// synthesis loop
unsigned long dur =(unsigned long)(duration*envoscil.GetSr());
```

```
for(unsigned long n=0; n < dur; n++){

    envoscil.DoProcess(); // envelope
    oscil1.DoProcess();   // oscillators
    oscil2.DoProcess();
    oscil3.DoProcess();
    oscil4.DoProcess();
    oscil5.DoProcess();
    oscil6.DoProcess();
    oscil7.DoProcess();
    oscil8.DoProcess();
    oscil9.DoProcess();
    mix.DoProcess();      // mix
    gain.DoProcess();     // gain attenuation
    output.Write();         // file output
                              }
  return 1;
}
```

Similarly, a user with some knowledge of C/C++ can easily build signal processing applications, just by writing his/her own `main()` functions and patching together the library objects. In addition, an experienced programmer would be able to extend the library by adding his/her own processing objects.

4 Sample applications

Together with the development of the SndObj library, a number of sample applications were developed with a twofold purpose: as a set of programming examples and as a group of useful programs to be used in computer music composition. They include command-line cutting, splicing and mixing programs, multi-purpose filters, reverbs, synthesizers and other utilities. It is expected that they will form a comprehensive set of tools for composers of electroacoustic and computer music. The technical documentation of the SndObj library will include the complete code for all the sample applications.

5 Graphical user interface

Studies are being carried out to couple the development of the library objects with a graphical user interface (GUI) using the V C++ framework (Wampler, 1996). V is a portable C++ GUI library developed for X-Windows and MS-Windows environments. Because of its portability across the platforms used in this research project, it has been considered for the implementation of visual counterparts to the objects of the Sound Object Library. These would be processing boxes that would provide an intuitive interface to the C++ code. The initial idea is to develop a patching

application that will let the user play with the sound objects to create his/her own signal processing module, by interconnecting the available boxes. At the present moment, some sample visual applications were developed using V and the SndObj library, running under MS-Windows and X-Windows (AIX and Solaris). These are intended as test applications for the development of a portable GUI. This would provide an intuitive use of the library objects and a powerful tool for composers of electroacoustic and computer music.

6 Conclusion and further research

This paper described the development of a set of objects for sound manipulation in the computer. The Sound Object Library is being designed to be a comprehensive multi-platform set of C++ classes to be used by programmers and composers. It has a modular design, akin to analog synthesizers and computer music systems, and simple connectivity. All the processes involved in the sound production, manipulation and storage are encapsulated by the library classes. Research is continuously being carried out in the development of new objects, including spectral analysis and resynthesis. The development of a GUI for the library, using the multi-platform V framework, is also under study. A beta-version of the first release of the SndObj library, together with command-line and graphic sample applications, will soon be available for download at the internet site http://www.dcop.uel.br.

Acknowledgments

The authors would like to thank CNPq for financial support of the present research and the Department of Computer Science (UEL) for granting the use of the IBM-RISC lab and the ULTRAsparc workstation.

Bibliography

Dodge, C & Jerse, T (1985). Computer Music: Synthesis, Composition and Performance. Schirmer Books, New York.

Lazzarini, VEP (1998). "A Proposed Design for an Audio Processing System ". *Organised Sound* 3 (1). Cambridge Univ. Press, Cambridge.

Moore, FR (1990). *Elements of Computer Music*. Prentice-Hall, Englewood Cliffs.

Stroustrup, B (1991). *The C++ Programming Language.* Addison-Wesley Publishing Co., Reading, Mass..

Rumbaugh, J ; Blaha, M; Premerlani, W; Eddy, F; Lorensen, W (1994). *Modelagem e Projetos Baseados em Objetos*, Editora Campus, Rio de Janeiro.

Vercoe, B (1992). *Csound, A Manual for the Audio Processing System.* MIT, Cambridge, Mass..

Wampler, B (1996). *V Reference Manual.* HTML document. http://www.cs.unm.edu/~wampler/vwebref.html

AN AUTONOMOUS STYLE-DRIVEN MUSIC COMPOSER

LUCIANO VIEIRA LIMA
Departamento de Engenharia Elétrica
Universidade Federal de Uberlândia (UFU)
Uberlândia – MG - Brasil
dreams@nanet.com.br

JOÃO JOSÉ NETO
Departamento de Engenharia de Computação e Sistemas Digitais
Escola Politécnica da Universidade de São Paulo (EPUSP)
São Paulo – SP – Brasil
jjneto@pcs.usp.br

Abstract

This paper resulted from our search for better knowledge of the human's mind creative processes, not only in music but also in other fields. Music is one of the most complex, abstract and intuitive activities executed by human beings. We believe that having success in this domain will help understanding and modeling phenomena involving creativity and temporal reasoning. Our approach has been musical composition based on existent musical styles, or modeling several composers' knowledge, creativity, styles and works. Our system's model composes music following the style of any composer, focusing mono- or polyphonic solo instruments. In a fashion similar to that used by many composers of pop-music, our work searches for solutions for this problem using a note-by-note approach, with no help of traditional rules or personal knowledge of musical theory. This article also discusses whether or not this task can be accomplished wit no user interaction and without backtracking procedures. Our composition system should start from musical inputs; form that all needed composing knowledge has to be extracted. This work illustrates mankind's need to leave to future generations not only results of work but also the creative process that originated them.

1-Introduction

Many works on this subject have been analyzed and discussed that use harmony rules, stochastic grammars and similarity methods in their creation process. For style-driven composition, reference composers' works are searched for elementary repetitive forms that identify their style. Such a system, presented by David Cope (Cope 1988, 89 e 92), uses the technique he calls EMI. In his work, Cope locates the smallest repetitive forms, grains (Smoliar 1991), in music, and associates them to verbs, objects, nouns, adjectives etc. The spaces that pass it action sensation are classified as verbs, and the variations that follow it, as objects, e.g., a music fragment that seem to be receiving some action is classified as a subject. Cope reorganizes subjects and verbs, forming new musical sentences, following

given grammatical rules. By changing weights, Cope interacts dynamically with the composition process, tuning the quality of the output with respect to their closeness to the rules of the given grammar. For that purpose, trial and error, as well as backtracking techniques are thoroughly used in the generation of the compositions.

2-Goals

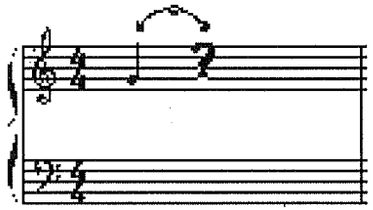
In contrast to similar systems published in the literature (Dodge & Jesse 1985; Hudak 1996, Todd 1989, Cope 1992), our system achieves several important goals:

1. it is able to generate new compositions with no interaction with the user
2. it does not require explicit rules of harmony and musical composition
3. it learns and generates music with no help of previous analysis
4. It is free of trial-and-error (Gogins 1991) procedures so all generated notes are correct.
5. Feedback of system-generated music do not change system's information base on style.

3-The model

Informally, a musical composition may be classified as pop music and classical music. Each piece of classical music in general shows its own well-defined, personal and explicit structure and semantics, in contrast to most pop music compositions.

The model presented in the present paper doesn't intend to capture and to generate new structures. Instead, given a structure, it models and simulates how each author links and generates musical notes in a musical sentence or theme. Once defined the desired structure, rhythm and musical harmony, our system starts to define the building grains of the composer's style from which it will build new musical compositions. In order to build each of these grains, the system starts from some initial note, and determines the following ones by answering successively the same question: which is the next note to be generated? (See figure)



This article presents for this question a proposal of answer that allow producing, at a low cost, quite satisfactory results.

4-Feelings

Great maestros have perhaps better musical knowledge than famous composers of the past do. However, most of them admit not to be able to compose comparably, for technical knowledge on composition techniques and harmonization are not enough to build good composers.

Thus, even by using well-known composition rules, and by well knowing and recognizing a given composer's style, the choice of how to use all this knowledge is always defined intuitively, taking into account cultural, social and emotional aspects of each one's life. So, each composer is permanently subject to dynamic unexpected events in his life, which may severely affect his or hers musical style (Lima 1998).

As an illustration, we can imagine a composer that fights his relatives has cut his account in the bank, has his dog run over, etc. Some time later, he composes a melody.

Many of these adversities will certainly have influence in the composition. These unexpected random facts are very hard to model.

As another example, we have a real fact occurred when the pop composer Eric Clapton lost his son in an accident. After a while, Clapton composed music in honor to his son, which registers that sad moment, and the deep suffering it induced. How to instruct the computer about the feeling of a son's loss? Anyone that has a son, or that lets himself to act as someone that has or that had one, realizes that it is very difficult to explain that in words, and far harder to model it.

Even for a human being, which never had a child, it is difficult to fully understand such a feeling. Thus, even for the composer himself, there are moments of inspiration, motivation and creativity whose replication is virtually impossible. From such unique facts, emerge several phases in a composer's career, although such phases are, also, strongly associated to the composer's technical evolution.

Yet regarding feelings, another fact also very hard to model is that the same person reacts differently to a single given stimulus in different circumstances. So, it is a complicated task to generate a system of automatic composition using computers embedding the ability to handle feelings, in an autonomous way, in the automatically generated melodies. So, how to model and to reflect in music great passions, sadness, joy, a disillusion? How to model feeling?

By now, this problem seems unsolved. In our system, we intend to reproduce or to simulate, in the generated music, part of the feeling expressed by composers in their music, no matter which these feelings are. This is a rather simpler task to perform. So, the choice of which note should follow a given one in a certain musical structure, is a problem for which solutions may be developed which are based solely on the contents of the music used as base, with no help of formal concepts and technical training. Being impossible to explain and to model the feelings of a composer, how to design a system that composes melodies reproducing them? We can only assure that undoubtedly some particular composer's feelings and emotions have been recorded in his music at the moment of its composition. Therefore, a system that composes based on the author's work only is expected to capture some of the moments and the original author's feelings from the original work and reproduce them, although in some imperfect form, in the generated melodies.

5-Our system

The system breaks music into 3 components: Rhythm, Harmonization and Melody.

Similarly to what occurs in many style-driven composition systems, our system tries to generate the smallest grain, by reproducing the activity of the composer executes when originating such grains, according to the method described below.

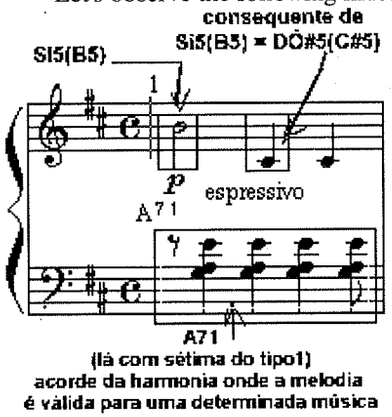
The system described in this paper has been implemented in two versions, both in functional languages: the first was built in LISP and the second, in language CLEAN, to be executed in IBM-PC-compatible computers.

5.1-The concept of consequent note

A consequent note is any note the system is allowed to generate after an already composed one. Our approach to determine possible consequent notes of a given one it that

the note has been used in similar circumstance in the compositions used as base. The key is defining the circumstances under which such similarity holds.

Let's observe the following illustration:



Converting the score's compass 1 to internal format gives:

1. (B5 1:2,C#5 1:4, C#5 1:4)

[At 7 1]; where 1:2 = half note and 1:4 = quarter note

Decomposing into the 3 components gives:

Rhythm = (1:2 1:4 1:4)

Harmony = (AT 7 1)

Melody = (B5 C#5 C#5)

Based on this information, and on the rest of the music, the system should be able to extract necessary data to generate new compositions. In the example notes and musical chords were given traditional names, but any name is allowed, since the system does not interpret them. Thus, note B5 could have been called, for example, note 1, chord A 7 1 could have been called chord 1 and so forth. At all occurrences of the same note or chord appear the same name is to be used.

Note by note generation, using the concept of consequent note is not as simple as it can seem. It first seems that the problem consists only in consulting the notes that the composer used in its music and those that follow it. So, any note that the author has used as consequent to the previous one it should be a correct note in a future composition. That is not true, and it can cause serious trouble in the automatic process of composition, as illustrated in the following example:

Suppose that, in a given melody, the author used do2, sol1 and la1 as consequent to note do1. If the author uses solely mi1 as a consequent of la1, and only la2 as consequent of mi1, and do1, sol2 and fa1 as consequent for sol1.

With these informations, the following information is collected by the system:

Consequent of DO1 = (DO2, SOL1, LA1)

Consequent of LA1 = (MI1)

Consequent of MI1 = (LA1)

Consequent of SOL1 = (DO1, SOL2, FA1)

This database supports an algorithm that randomly generates the first note, and automatically determines a consequent for that note, and so forth. This way, a possible composition generated automatically by such algorithm could be: (SOL1 - DO1 - LA1 - MI1 - LA1 - MI1 -... - repeating indefinitely the sequence LA1 - MI1).

Here the algorithm fails, because, when generating a note that has an only recurrent sequence of consequents, the system starts to repeatedly generate that sequence of musical notes. In the algorithm it should exist, something else, allowing the generation of the next note, whatever would be the data resultants of the simple collection of the consequent notes

that the author used in his original compositions. So, only the notes of its melody should not analyze the original music. Rhythm and harmony decisively affect the choice of consequent note, and should also be considered. Even when the original music is monophonic, harmony is decisively embedded in the composed melody.

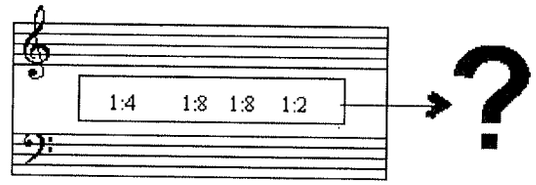
Avoiding recurrent sequences of notes is another point that distinguishes our system from many similar works. Our solution is based on composition by compasses. As each compass carries together information on rhythm, the closure of its metric and the solution for its harmony, each compass may be considered a basic self-contained element inside the final composition.

5.2-The algorithm

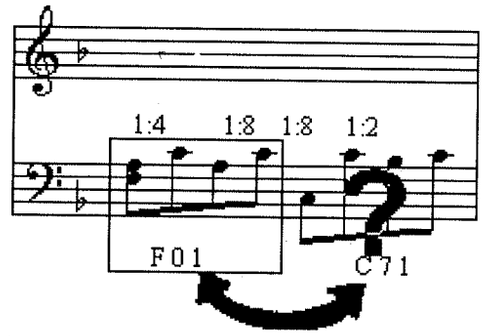
For implementing the composition routines, we must first create information bases and histograms, that allow translating all events in a music, linking them tightly in order to preserve the maximum of information that enables producing new music, with a style as close as possible to the music used as base of the adopted model.

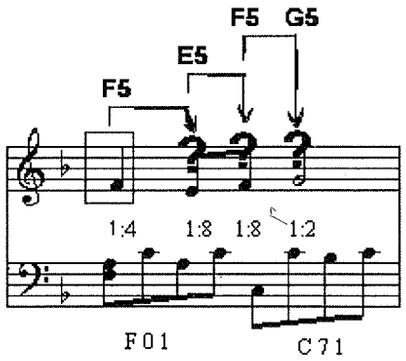
Three basic histograms are created with that purpose:

· **RHYTHM HISTOGRAM** - it indicates, for each compass, which rhythm can follow a given one. That information is calculated with base on the probability that, in similar conditions, the original composer would choose each of these compasses as a consequent.



CHORDS HISTOGRAM - This information allows the system to compose harmony for the generated music, by indicating which chord type can follow a preceding one, with base on the rhythm of the current compass, its antecedent and its consequent, and observing, also, the probability that, in similar conditions, the original composer would choose some chord as a consequent for the previously generated one.





· **NOTES HISTOGRAMS** - This histogram allows the system to compose the melody, based on the rhythm and harmony of the current, preceding and succeeding compasses, statistically indicating with support of similar situations observed in the author's original works, which notes should follow a given one.

The composing algorithm

After building histograms, the composing algorithm is naturally divided in three different strongly interconnected stages: the Rhythm stage, the Harmony stage and the Melody stage. A distinguished features in this system is that, in the note-by-note generation process, consequent notes are determined by the rhythm and especially by the harmony in each compass.

5.2.1-Composing rhythm

Choose the type of musical style (rhythm) or an author's music.

1. Determine the number of compasses for the new music.
2. Randomly choose a type for the first compass or choose a starting compass for the music.
3. Randomly choose a type for consequent compasses, according to the obtained compass histogram. Randomness is assured by the manner histograms are built by the system.
4. Repeat step 4 until all compasses determined in item 2 are built.

Example for three compasses:

1. (1:2, 1:2)
2. (1:2) (1:2)
3. (1:4, 1:4, 1:4, 1:4)

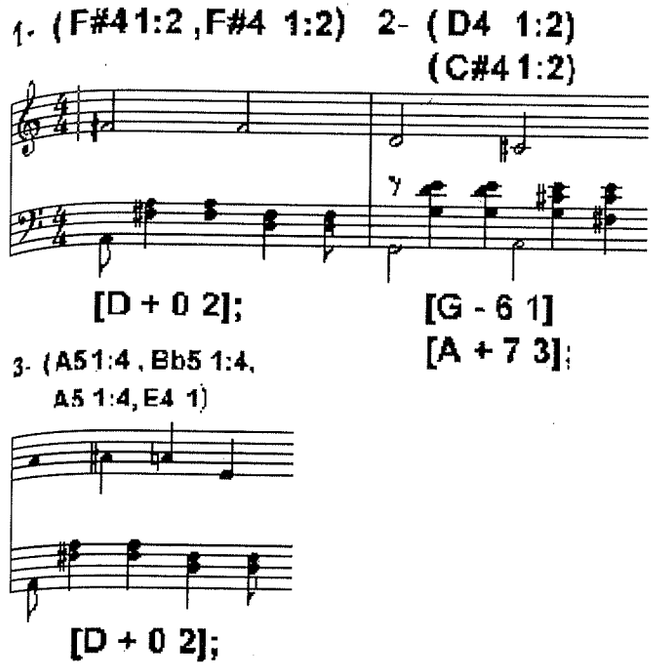
So, compass 1 will have two notes whose duration is of a minim (1:2). The compass will have just one chord. It indicates that compass 2 will have, also, two notes with duration of a minim each, but this compass will have two chords. Finally, the third compass indicates that the compass will have four notes with duration of a quarter note (1:4) and a single chord.

5.2.2-Composing harmony

1. Randomly choose the first chord of the current compass.
2. Randomly choose a consequent chord for the current compass. In the case the chord fills the whole compass, choose randomly a chord for the consequent compass.
3. Repeat item 2 until all chords in all compasses are formed.

Example for three compasses (continued):

1. (1:2, 1:2) [D + 0 2]
2. (1:2) [G - 6 1] (1:2) [A + 7 3]
3. (1:4, 1:4, 1:4, 1:4) [D + 0 2]



That means compass one has one chord [D + 0 2] driving the generation of two notes, the second compass has two chords [G - 6 1] and [A + 7 3], driving the generation of one note, and the third compass has one chord [D + 0 2], which drives the generation of four notes.

5.2.3-Composing melody

1. Randomly choose the first note for the first chord of the first compass
2. If that is the last note in the chord, choose randomly its consequent transition note from the current chord to its consequent chord, otherwise, choose a consequent note for the chord randomly.
3. Fill all notes in all compasses repeating item 2 for all chords in the music.

Example for three compasses

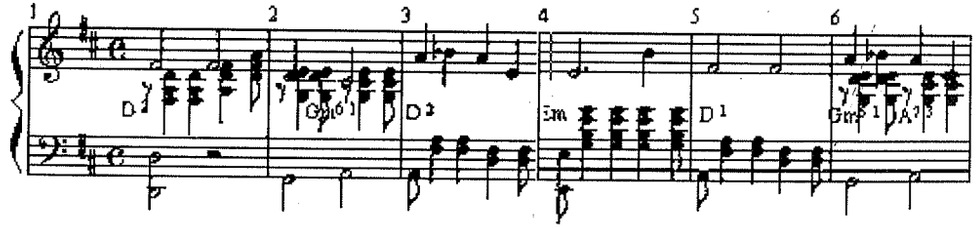
(final):

1. (F#4 1:2, F#4 1:2) [D + 0 2]
2. (D4 1:2) [G - 6 1] (C#4 1:2) [A + 7 3]
3. (A5 1:4,Bb5 1:4,A5 1:4,E4 1:4)[D + 0 2]

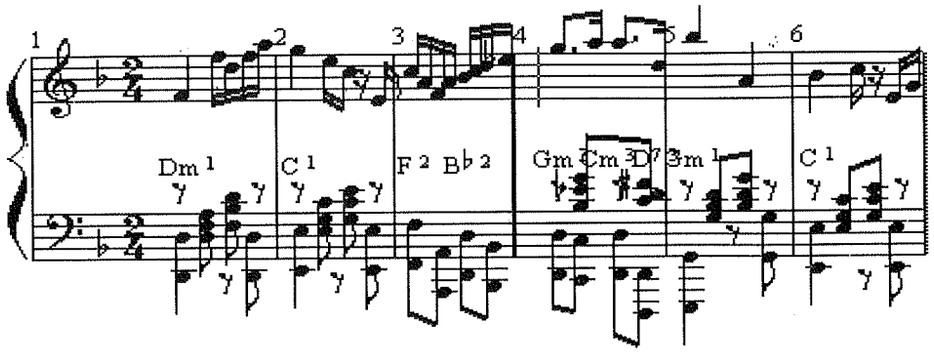
Translating into a conventional score (CPN) leads to the score above.

5.3-Fragments of music, composed by the system based on classic authors:

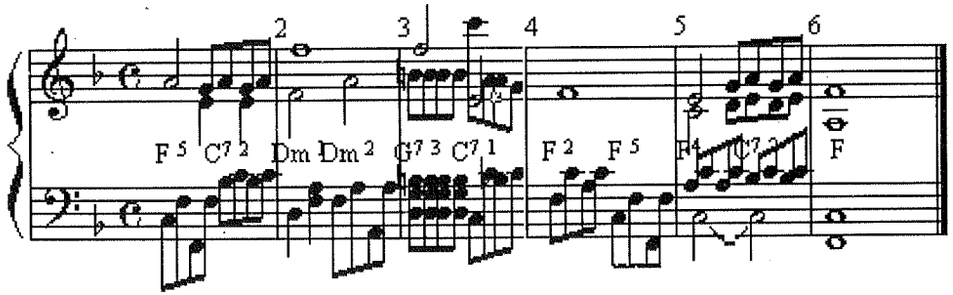
1. Six compasses based on None but a Lonely Heart by P. Tchaikowsky



2 - Six compasses, based on the Symphony of the Cantata 156 by J. S. Bach



3 - Six compasses based on Tristesse by F. Chopin.



Note that our system uses neither trial and error nor backtracking procedures; it can learn and generate music just analyzing existent music; Its base of information remains unchanged after being feedback with generated music; it uses no explicit formal harmony or composition rules.

Our system also differ from other systems based on note-by-note composition, for applying composition concepts on musical compasses and for using harmony (chords) in the decision process while choosing consequent notes.

One may ask: if our system adds nothing to the rhythm, chord and melody histograms for a given author, how would new music be generated?

Creating new music in a given style should not be interpreted as creating new music in a new style. Our system was not designed to make any improvements or to set modifications to an author's original style. While other systems accept musical intervals of a sequence, such as - la - do - mi - as well as mi - do - la , if a given author just used the first sequence in his compositions, accepting the other sequence may alter the style we are trying to follow.

As an analogy, one may analyze a piece of text whose author always uses sentences in direct speech. When doing automatic composition, using reversed speech in the generated phrases would depart away from the original style.

Vocabulary is to be preserved, as well as the general form of the sentences, which must follow original writer's phrases in their grammatical structure and style.

Similarly, the system presented in this article creates new melodies with no change to the original histograms, in contrast to other style-driven systems.

Our method for determining consequents may also be compared to a picture-generating system: such a system would build human pictures in which we would find more than one nose, mouths and ears would be wrongly placed, etc. This really happens, since no grammar is supplied to define the structure of the work to be generated. Instead, as it is implemented by now, our system was designed to generate those parts - noses, mouths, necks, etc, from given models extracted from existent work. In this case, the system tries to generate its output following as close as possible the style of the artist whose work is being used as model.

6-Future work

Presently a research is in progress, in order to improve our already developed tool by eliminating one of its fragilities, the absence of structure in the implemented generation algorithms. As it has been conceived to generate basic elements of a music, the external consequence of using it to generate a whole music allows one to easily identify the mentioned fragility in the resulting output, which, although being very pleasant in its melody, metrics and harmony, undoubtedly do lack global structure.

The way algorithms were developed, it is assumed that melody histograms used by the system to generate music are extracted from significant samples of original pieces, and that they are representative of music fragments of the same type.

Current work is being developed to extend the generating program in order to convert the existing one into a module of a larger system, which will have an external control section designated to handle structure.

In order to accomplish that goal, our project will take place in several phases; each one giving rise to a product representing some significant progress when compared to the previous one:

6.1-Phase 0 - in the current stage, we have a generator that is able to compose homogeneous fragments of music, starting from samples of original fragments assumed to be of the same nature. No connection is allowed among generated fragments, and there are no tests on inadequate classification of the fragments used as model. The practical output of the currently available implementation is in general pleasant but structure music.

6.2-Phase 1 - in progress - a natural progress to the previous phase consists of allowing the user to provide samples of several kinds as models. Users may then request to the system that composes a sequence of fragments following some user-specified order. The system then generates and links the composed fragments through appropriate passage sequences. The result is a sequence of properly linked fragments that follow a structure manually imposed by the user. There will be no tests on the imposed structure or on the criteria used to classify the given samples.

6.3-Phase 2 - to be initiated next - in this phase, a first automatic procedure will be included for generating the structure of the generated music. Users should manually replace the system's previous fixed structure with some structuring rule. The system will then apply that rule to generate all needed different structures of each kind, following the supplied rule.

Usual sentence derivation from a grammar will automatically generate the desired structure for the generated music. The user may impose restrictions, e.g. on the maximum number of fragments, of fragments of a given type, of repetitions of certain sequences of types. The expected result is the automatic generation of music with sound similar in quality to the previous ones, and showing the structure created by the system under user's specification.

6.4-Phase 3 - to be started in short term - the next stage refers to obtaining structuring rules automatically. The underlying grammar of the structure imposed to automatically generated compositions will be itself deduced by the system. A musician must help in this activity by supplying structural analyses for a significant set of works to be used as models, identifying and classifying the fragments in each work. Then, an algorithm is executed to make a grammatical inference that generalize the information extracted from the given samples, in order to infer a formation rule to be used as a grammar describing the desired structure. This phase will result in a complete synthesis program, for music will then be obtained automatically both in contents and in formal structure. User's participation will be restricted to inserting information on the structure of the samples used as models.

6.5-Phase 4 - this phase is foreseen to begin in some years. It is much more complex, because it should include heavier artificial intelligence resources; in order to operate with no human-supplied information concerning the structure of the pieces used as models. Structural analysis should be made by programs with enough built-in knowledge on musical analysis to allow automatic identification of the musical elements present in the score. This phase will produce software that will be able to perform musical analysis and synthesis, without direct participation of the user.

7-References

COPE, DAVID. **Experiments Musical in Intelligence (EMI): Non-linear Linguistic-Based Composition** Interface, vol, 18, p. 117-139, 1989.

COPE, DAVID. **Computer Modeling of Musical Intelligence in EMI**. Computer Music Journal, vol 16, no. 2, Summer, p. 69-83, 1992.

COPE, DAVID. **Music & Lisp**. OH Expert, March, 1988, p.26-33.

DODGE, CHARLES; JESSE, THOMAS A.. **Composition With Computers - Computer Music - Synthesis, Compositions and Performance.** NY, Schirmer, London, Coller MacMilan, 381 pg., p.265-321, 1985.

GOGINS, MICHAEL. Interated Functions Systems Music. Computer Music Journal, vol. 15, no. 1, Spring, 1991, p.40-48

HUDAK, PAUL; MAKUCEVICH, TONE; DDE, SYANGA; WONG,BO.**Haskore Music Notation–An Algebra of Music** -. Journal of Functional Language, vol 6, May, 1996 p.465-483

LIMA, LUCIANO VIEIRA. **Um Sistema de Composição Musical Dirigido por Estilo.** Tese de Doutorado, Universidade de São Paulo, São Paulo - Brasil, 1998.

SMOLIAR, STEPHEN W. Book Review and Response - **Models of Musical Communication and Cognition**. Interface, vol. 18, 1990, p. 361-371

TODD, PETERSOM. **Connectionist Approach to Algorithmic Composition**. Computer Music Journal, vol. 13, no.4, Winter,1989, p.27-43.

SOUND FUNCTORS APPLICATIONS

JÔNATAS MANZOLLI[2], ADOLFO MAIA Jr.[1,2]
Mathematics Department[1]
Interdisciplinary Nucleus for Sound Communications (NICS)[2]
University of Campinas (UNICAMP) - Brazil
Jonatas@nics.unicamp.br, adolfo@nics.unicamp.br

ABSTRACT

The concept of mathematical structures called Functors can be useful to develop a large number of compositional algorithms. We introduce here concepts such as Categories and Functors as generalised sound construction tools. We define two applications: a functor between plane curves and sound events based on the MIDI protocol and a functor between the class of continuous curves $C(x)$ *defined in a finite interval* $I \subset \Re$ *and the class* Ψ *of Fourier Spectra* $A(v)$.

INTRODUCTION

Music composition is plenty of procedures that can be related to mathematical symmetries, it is presented in (Hofstadter 1989), a study relating music, design and mathematical structures. On the other hand, music symmetries can be better understood and handled if we search for underlined or hidden mathematical structure that generates them. In this work we show that the intuitive use of symmetry and associations in music can be studied and explored through the mathematical concepts of Category and Functor.

This approach is interesting, not only because it can be used to classify sound material, but mainly for it furnishes new relations pointing out to an enormous variety of sound generation methods. In short, we claim that Functors, in the same way it was firstly devised in mathematics (Maclane & Birkohoff 1953, 1979; MacLane 1971), can be generalised as universal tools to construct mathematical models for music composition and sound synthesis.

There has been a series of approaches applying mathematics to build sound structures such the use of 1/f noise fractal distribution (Voss & Clark 1978; Bolognesi 1983), non-linear dynamical systems and iterated function systems (Pressing 1988; Scipio 1990; Gogins 1991). There is a study about these systems in (Manzolli, 1993a).

Our group has been worked with applied mathematics to produce sound design machines focusing new methods for sound synthesis using non-linear dynamics (Manzolli 1993b), mathematical models for algorithm composition such as Markov Chains and

Boundary Functions (Manzolli & Maia 1995) and development of interactive desktop and gesture interface for composition in real time (Manzolli & Ohtsuki 1996).

Below, we discuss Functors as generalised Algorithmic Compositional tools for sound construction defining Class, Morphism, Category and Functor. It follows applications in the sound domain.

CATEGORIES AND FUNCTORS

Functor is a kind of function, as suggested by its own name, but it is not an ordinary one because it carries the underlined structures of the sets in which it is applied. These sets are called Categories. More precisely, a **Category** Φ is defined by three proprieties:

a) A *class of objects* **A, B, C...**

b) For each pair of objects $\mathbf{A, B} \in \Phi$ we have a set of applications (morphisms) **M(A,B)** from **A** to **B**.

c) For each triple of objects $\mathbf{A, B, C} \in \Phi$ we have a composition law for the morphisms

$$\mathbf{M(A,B) \times M(B,C) \rightarrow M(A,C)}$$
$$\mathbf{(f,g) \longrightarrow g \circ f}$$

Now, for the proprieties above it follows three axioms, which are primary properties of these categories:

$\mathbf{A_1}$) The sets of morphisms **M(A,B)** and **M(C,D)** are mutually disjoints unless $\mathbf{A = C}$ and $\mathbf{B = D}$.

$\mathbf{A_2}$) Associative Law: $\mathbf{h(gf) = (hg)f}$.

$\mathbf{A_3}$) Existence of Identity: for each object **A** there exists a morphism identity $\mathbf{1_A}$: $\mathbf{A \longrightarrow A}$ such that for any $\mathbf{f: A \longrightarrow B}$ and $\mathbf{g: C \longrightarrow A}$ we have $\mathbf{f \circ 1_A = f}$ and $\mathbf{1_A \circ g = g}$.

Usually, the theory of categories and functors is mathematically involved. Here we use only basic properties of the definition above to show that underlined structures can be mapped from categories of mathematical objects to categories of sound objects and the last undertake formal properties of their mathematical counterparts via a functor.

Given two categories Φ and Ψ, a functor **F** between Φ and Ψ is a map which associates each object $\mathbf{A} \in \Phi$ to an object $\mathbf{F(A)} \in \Psi$

$$\mathbf{F: \Phi \longrightarrow \Psi}$$
$$\mathbf{A \longrightarrow F(A)}$$

and for each morphism $\mathbf{f} \in \mathbf{M(A,B)}$ associates a morphism $\mathbf{F(f)} \in \mathbf{M(F(A), F(B))}$ with the properties

$$\mathbf{F(gf) = F(g)\,F(f)}$$
$$\mathbf{F(1_A) = 1_{F(A)}}$$

Functor **F** operates on morphisms as well on elements of the category Ψ. In this way, the structure of a product (or composition) between two morphisms in the category Φ is transported to morphisms in the category Ψ via functor **F**.

There are too many examples of mathematical functors. Since we are focusing music application only, two simple examples of functors applied to sound categories are presented in the next section.

APPLICATIONS

EXAMPLE 1 - **Sequence of MIDI Events**

We begin with the mathematical category Φ = {***continuous finite curves in*** **a bounded region** $\mathbf{U} \subset \Re^2$}. Given two curves $\mathbf{C_1}, \mathbf{C_2} \in \Phi$, a morphism in Φ as an application

$$\mathbf{f: C_1 \longrightarrow C_2}$$
$$\mathbf{x \longrightarrow f(x)}$$

which means to deform $\mathbf{C_1}$ into $\mathbf{C_2}$, i.e. $\mathbf{f} \in \mathbf{M(C_1, C_2)}$. The product **gf** is defined as the composition $\mathbf{gf = g \circ f}$

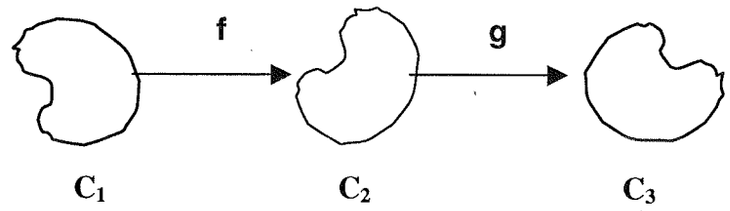


Figure 1. Composition of Plane Curves

We define Ψ as a category of functions to control sound parameters described by the MIDI (Music Instrument Digital Interface) protocol. Thus, $\mathbf{F(f)} \in \mathbf{M(F(C_1), F(C_2))}$ and **F(f)** deforms $\mathbf{F(C_1)}$ to $\mathbf{F(C_2)}$. Given a curve **C** in Ψ, the function $\mathbf{F(C)} \in \Psi$ can be defined in several different manners. Particularly, **F(C)** can be expressed by the Distance Function between a fixed point in the plane to the points of the curve **C**. It is easy to see that the functor property is satisfied, namely

$$\mathbf{F(g(f(C))) = F((gf)(C)) = F(g)F(f)(F(C))}$$

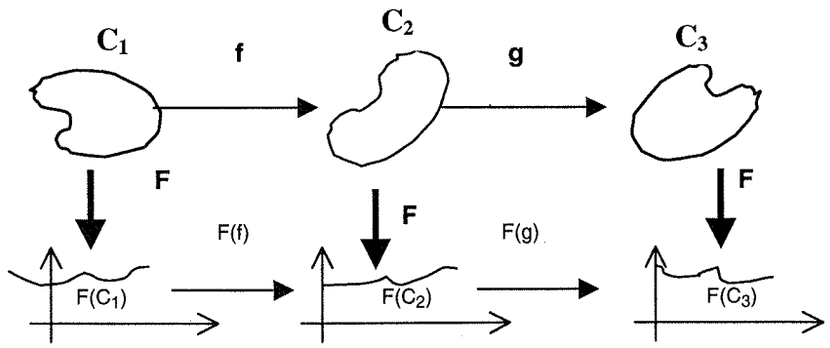


Figure 2. Diagram of the Functor operation

We have implemented this functor of plane curves to MIDI parameters in a computer program called CurvaSom. This software uses a graphic interface running in a MS Windows NT environment. Here we are just introducing main theoretical aspects of the Sound Functor approach.

EXAMPLE 2 - **Sound Synthesis**

In this example, an application of a functor is used to build Spectral Envelops. We take as mathematical category, a class Φ of smooth functions $\mathbf{C(x)}$ defined in a finite interval $\mathbf{I} \subset \Re$ and as sound category, a class Ψ of Fourier Spectra $\mathbf{A}(\nu)$ of a finite set of sounds. Here the function $\mathbf{C(x)}$ acts as a shaper of a spectral set. Starting from a fixed spectrum, which we call Input Spectrum, we use deformation of curves as morphisms in both classes of functions. We construct the following Functor:

$$\mathbf{F}: \Phi \longrightarrow \Psi$$
$$\mathbf{C} \longrightarrow \mathbf{F(C)} = \mathbf{C(A(\nu))}$$

Where the parameter $x = A(\nu)$, with $0 \leq x \leq A_{max}$ and A_{max} is the maximal amplitude considered.

The morphisms in Φ are deformations of smooth functions and the morphism in Ψ can be chosen as $\mathbf{F(f)=f}$. This means that the morphisms in Φ are used as spectral deformations. Using the above definition it is easy to see the property $\mathbf{F(g)\ F(f) = gf = F(gf)}$ is satisfied.

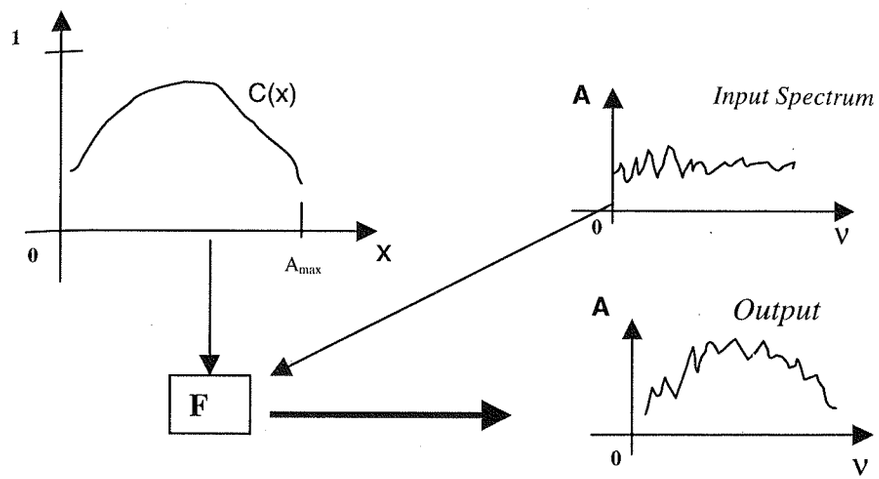


Figure 3. Spectrum Functor Diagram

CONCLUSION

We can construct several examples similar to the above ones. Another example using a different mathematical construction is presented in (Maia, Valle & Manzolli 1998). Other mathematical structures like groups, lattices, algebras, several different geometrical and topological structures can be used as mathematical categories (see MacLane 1971) in order to get sound outputs through a suitable choice of functors. Functors allow us to construct a huge quantity of sound outputs reflecting the structure (or symmetry in several cases) of external mathematical categories used to generate them. Musicians who work on Computer Music, aware of the properties of functors, can expand their sound tools in a way, certainly, not yet explored by the authors of this paper.

Mathematical tools in composition brings new possibilities for composers envision development of Compositional Systems. Research in Computer Music provides powered tools for constructing these systems. This union of artistic and mathematical knowledge creates a framework for investigation and music production, an environment for applying mathematics in order to manipulate sound structures. Mathematical models presented here could be expanded using graphic interfaces to create new musical performance situations, making mathematical design to produce computer music instruments.

REFERENCES

Bolognesi, T. (1983). Automatic Composition: Experiments with Self-Similar Music. *Computer Music Journal* 7(1):25-36.

Gogins, M. (1991). Iterated Functions Systems Music. *Computer Music Journal* 15(1):40-48.

Hofstadter, D. R. (1989). *Gödel, Escher, Bach: an eternal golden braid.* Vintage Books, New York, ISBN 0-394-75682-7.

MacLane, S. (1971). *Categorias for the working mathematician.* New York, Springer-Verlag.

MacLane, S. & G. Birkhoff (1953). *A Brief Survey of Modern Algebra.* New York, the MacMillan Company.

MacLane, S. & G. Birkhoff (1979). *Algebra.* New Yok, Macmillan Company.

Manzolli, J. (1993a). Non-linear Dynamics and Fractals as a Model for Sound Synthesis and Real Time Composition. PhD Thesis at the University of Nottingham, England.

Manzolli, J. (1993b). *Musical Applications Derived From FracWave Sound Synthesis Method. Proceedings of the 94th Audio Engineering Society Convention*, Berlin.

Manzolli, J. (1994). FracWave: Non-linear Dynamics as Timbral Constructs. Procedings of the *of the XIV Congress of the Computer Brazilian Society, I Symposium on Computer Music,* Caxambú - Brasil.

Manzolli, J. & A. Maia Jr (1995). Interactive Composition Using Markov Chain and Boundary Functions. *Procedings of the XV Congress of the Computer Brazilian Society, II Symposium on Computer Music*, Canela, Brazil.

Manzolli, J. & W. Ohtsuki (1996). INTERASOM: a desktop for Algorithmic Composition. *Procedings of the XVI Congress of the Computer Brazilian Society, III Symposium on Computer Music,* Recife, Brazil.

Maia Jr., A., R. do Valle & J. Manzolli (1998). Estruturas Matemáticas como Ferramenta Algorítmica para Composição. Submitted to the XI Congress of the Brazilian Association for Research in Music, ANPPOM, University of Campinas.

Pressing, J. (1988). Non-linear Maps as Generators of Musical Design. *Computer Music Journal* 12(2):35-45.

Scipio, A. (1990). Composition by Exploring of Non-linear Dynamic System. *Proceedings of the 1991 International Computer Music Conference.*

Voss, R. F. & J. Clarke, (1978). 1/f noise music: Music from 1/f noise. *Journal of the Acoustical Society of America* 63(1):258-263.

ACKNOWLEDGMENT

We thank Prof. Raul do Valle for fruitful discussions and the Foundation for Research of State of São Paulo - FAPESP for supporting this work.

Busca e Recuperação de Informação Musical

Ana Miccolis
Universidade Federal do Rio de Janeiro
Coordenação dos Programas de Pós-Graduação em Engenharia - COPPE/UFRJ
Programa de Engenharia de Sistemas e Computação
Caixa Postal 68 511- CEP 21945-270 - Rio de Janeiro - RJ - Brasil
E-mail miccolis@cos.ufrj.br

Abstract

The process of musical composition can be benefit by the use of resources extracted from a database of musical passages. For this purpose, the composer may have access to the sound resources and be able to manipulate them in the compositional process. The traditional musical representation employed at the conventional music compositions allow the composer to retrieve the necessary information, to process it and to restore it in the same representation. The musical data in non conventional composition can't be represented only by the conventional written. In this case, the musician needs an resource that let the composer select music passages based on musical characteristics. One of the goals of information retrieval of musical data is to aid the research in compositional resources, like selection of timbre in a database. The database with this capability might provide the access to all the conventional attributes like alphanumeric data and the musical data. This paper investigates this kind of search and retrieval and which criteria one may consider at the musical information retrieval process.

1 - Introdução

Sistemas de recuperação de informação processam requisições de informações sobre arquivos identificando e recuperando destes certos registros que atendem aos requisitos exigidos. A recuperação de alguns registros em particular depende da similaridade entre estes e as consultas. Esta similaridade pode ser medida através da comparação de valores de certos atributos dos registros com os valores requisitados pela consulta. Em muitos sistemas de gerenciamento de banco de dados os arquivos contém registros homogêneos, com um conjunto de atributos especificado para caracterizar cada item do arquivo e com valores dos atributos que conseguem descrever univocamente e de forma completa os registros armazenados. Nestas circunstâncias a recuperação de informação depende de uma comparação exata entre os valores do arquivo e os da consulta. O nome de um autor ou intérprete, a data de nascimento, a sua nacionalidade e os títulos das obras musicais são tratados como dados não ambíguos e o processo de recuperação baseado nestes critérios é

simples. Contudo outros tipos de informação exigem a construção de modelos abstratos para representá-los e linguagens avançadas para definição e manipulação de dados. Num sistema de informações musicais, a recuperação de dados convencionais é importante, mas carece de outros elementos que auxiliem o músico no processo de composição ou análise. Seria de interesse para este tipo de usuário que além da seleção de obras a partir de atributos convencionais(autor, título da obra, instrumento de execução, etc.), a busca por trechos ou amostras sonoras pudessem ocorrer a partir da comparação entre elementos característicos do dado sonoro. A busca e recuperação poderia ocorrer em três níveis de abstração:

- Busca e recuperação de um trecho específico - Equivalente à comparação exata entre os dados de uma obra armazenada e os de uma consulta formulada. Poderia retornar por exemplo, uma Sonata de Mozart que possuísse no corpo da obra o trecho musical oferecido como àrgumento de pesquisa.

- Busca e recuperação por atributos acústicos mensuráveis - Equivalente à comparação em textos utilizando técnicas Fuzzi para selecionar trechos musicais que possuam atributos acústicos mensuráveis como altura, duração e volume, dentro de uma faixa de aceitação.

- Busca e recuperação por propriedades subjetivas do som - Correspondente à comparação entre elementos determinantes do timbre de um som como o envelope de amplitude, harmonicidade e envelope espectral.

Para atender às necessidades deste tipo de usuário, com conhecimentos de música e até de acústica musical, um sistema de informações musicais deveria oferecer busca e recuperação de informação de áudio em vários níveis de abstração. Os níveis 1 e 2 que são comuns à busca de texto e o nível 3 que é específico de áudio.

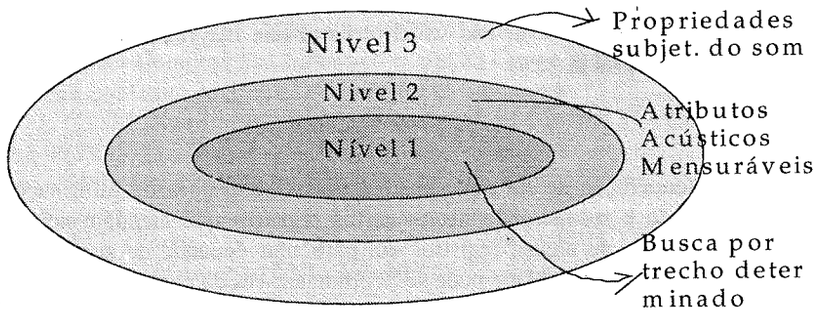


Figura 1 – Níveis de abstração de uma consulta em um sistema de informações musicais

Um sistema de informações musicais deve atender a diversos tipos de demanda de informação permitindo o acesso de usuários com diferenciados níveis de conhecimento no assunto. A grande quantidade de dados obriga o desenvolvimento de métodos de filtragem, seleção e interpretação dos dados que possibilitem um real aproveitamento de toda a informação disponibilizada. O objetivo da busca e recuperação de informações musicais é

prover diferentes níveis de acesso aos interessados em música. Entre os fins profissionais pode-se citar a pesquisa de recursos para composição (banco de timbres, por exemplo), o auxílio à pesquisa musicológica, transformação de exemplos sonoros e a possibilidade de busca de música não convencional através de uma representação própria. Para os diletantes um objetivo seria o de propiciar o conhecimento gradativo do universo musical disponível a partir de características próprias fornecidas por este usuário.

2 - Descrição dos Atributos da Informação Musical

2.1. A Altura

Na música convencional a identificação da altura de um som pode ser bem definida a partir da freqüência da fundamental, mas na música não convencional este conceito sofre interferência de outros parâmetros, que passam a ser manipulados neste tipo de composição. Experiências de Schaeffer(descritas no Solfejo do Objeto Sonoro) apontam para uma nova noção de altura na qual a compreensão de outros fatores torna-se tão ou mais importante que a medida de freqüência dada pela fundamental. Em seu trabalho, a exclusão da fundamental e a permanência dos demais parciais de um som possibilita a correta reconstrução da altura original. A possibilidade de supressão de harmônicos somente foi possível na música contemporânea e por esse motivo, as definições clássicas deste atributo começam a ganhar outros conhecimentos anteriormente desconsiderados.

2.2. A Intensidade

A intensidade traduz as sensações de forte ou fraco de um mesmo som e depende da amplitude sendo medida em decibéis. Outros termos que exprimem a noção de intensidade são a amplitude, volume, loudness e nos equipamentos midi, a velocidade. Segundo alguns autores a intensidade é influenciada não somente pela amplitude mas também pelas características temporais e espectrais do som (Moore 1989; Pierce 1992). O compositor pode empregar este recurso como forma de destacar um som, dando lhe mais intensidade que aos demais. Este recurso é muito utilizado para formar a dinâmica da música permitindo criar planos sonoros distintos, uma vez que ele está relacionado com a percepção de distância.

2.3. A Duração

Em geral na escrita convencional, a música é disposta em compassos de duração determinada pelo andamento da obra, o qual define a duração da unidade de tempo de um trecho musical. Para isso o autor da obra seleciona a unidade com que irá trabalhar(semínima, colcheia, etc.) e quantas unidades de tempo deseja que estejam contidas num único compasso. Na música não convencional, onde parâmetros como ataque de um som, podem ser manipulados, verifica-se que a noção de tempo pode ser alterada em função deste recurso. Invertendo-se o ataque de um instrumento, por exemplo, pode-se conseguir alterar a percepção de duração do mesmo. Isto acontece quando a parte final da duração do

som, que teria uma consonância conhecida do ouvinte passa a ter uma inversão. Desta forma o observador não se desliga da conclusão do som, passando a perceber a duração do mesmo como mais longa que a do mesmo som sem a inversão.

2.4. O Timbre

O timbre é a qualidade do som que permite a diferenciação entre os diferentes instrumentos musicais. Ele pode ser um valioso recurso para as composições contemporâneas, mas sua utilização ainda é dificultada pelo fato deste atributo ser fisicamente difícil de descrever e usar. Os primeiros estudos significantes nesta área foram realizados no século XIX por Hermann Helmholtz(1863) apontando para uma definição de timbre que o considerava apenas em função da composição de seus harmônicos. Depois da primeira síntese digital do som de Max Mathews em 1958 ficou aparente que o timbre não poderia depender apenas de distribuição harmônica do som, mas também de características temporais. Outras pesquisas de Risset e Matheus revelam que a proporção de harmônicos em certas classes de instrumentos musicais aumenta à medida que a intensidade cresce, dando início a uma nova abordagem do timbre na qual não se pode descrevê-lo por valores instantâneos de parâmetros numéricos, mas sim pela evolução deles, sugerindo que características espectrais e temporais são inseparáveis para definição do timbre. Nos estudos de John Grey(1977) ele aponta para uma solução tridimensional para a percepção de timbre. Uma interpretação física desses três eixos foi feita a partir da distribuição de energia do espectro harmônico, assim como das propriedades temporais dos sons. Em (Hourdin 1997) uma nova abordagem para o estudo do timbre é utilizada, aplicando técnicas multidimensionais à descrição física do som, com recursos do software Csound (Vercoe 1992). Neste trabalho a representação do timbre inclui tanto características espectrais como aspectos temporais do som permitindo uma classificação por ordem de importância das propriedade físicas. A vantagem desta abordagem reside principalmente na similaridade com que os tons são percebidos pelos humanos. A percepção de timbre aparece diretamente associada com a trajetória do som no espaço físico. Pode-se considerar para efeito de se extrair as componentes significativas do timbre, as características como envelope de amplitude e envelope espectral. A utilização da fórmula denominada "Fast Fourier Transform"(FFT) possibilita a observação do espectro harmônico de um som. O timbre sofre alterações durante o tempo de duração de uma nota. Por exemplo, se uma nota tocada no oboé iniciar com intensidade fraca e for progressivamente crescendo em intensidade o espectro harmônico irá se modificar com o volume. Novos harmônicos são adicionados ao som e ele ganha mais volume. Este é o motivo pelo qual em um som eletrônico não se consegue um efeito de "crescendo" convincente apenas aumentando a sua intensidade. O ataque de um tom é freqüentemente essencial para a identificação do timbre. Timbres com uma mesma distribuição de energia no espectro tendem a ser considerados similares. Em (Blum 1996A) o timbre pode ser analisado a partir de seus componentes: envelope de amplitude, harmonicidade e envelope espectral. O envelope de amplitude traduz a característica acústica de brilho. O brilho de um som é calculado como a centróide do espectro de magnitude da STF(Short-time Fourier). Exemplificando, quando coloca-se a mão sobre a boca, o brilho do som é reduzido

e também a sua intensidade. O envelope espectral traduz a característica acústica conhecida como largura de banda que é calculada como sendo a média ponderada da magnitude das diferenças entre os componentes espectrais e a centróide. Exemplificando, uma simples onda senoidal tem uma característica de largura de banda equivalente a zero, enquanto um ruído branco tem um valor infinito para esta característica. A harmonicidade permite a distinção entre um espectro harmônico(como vozes e sons musicais), um espectro inarmônico(como sons metálicos), e ruídos(espectros que variam aleatoriamente em freqüência e tempo). Ela é calculada medindo-se o desvio do espectro de um som de um espectro harmônico perfeito.

3 - Estrutura para Armazenamento dos Atributos do Som

A informação sonora deve poder ser recuperada a partir de similaridade acústica, sendo necessário assim medir esta similaridade. Alguns métodos tratam a busca de dados sonoros independente de suas características particulares, dando-lhes o mesmo tratamento utilizado na recuperação de imagens (Foote 1997). Contudo a pesquisa de timbres não consegue ser corretamente atendida por estes métodos, em virtude da priorização das alturas sonoras como fator de indexação. O timbre ao contrário das demais características acústicas deve ser definido a partir das suas componentes como foi visto no tópico 2. A pesquisa sobre a composição do timbre considera a análise de vários componentes, entre eles, o envelope de amplitude, o envelope espectral e a harmonicidade. Para permitir a recuperação de dados sonoros a partir de diferentes critérios, deve-se primeiramente medir uma variedade de atributos acústicos para cada som. Este conjunto de "N" atributos pode ser representado em uma estrutura "N-vetorial". Em um sistema de busca e recuperação de informações de áudio, o usuário deve poder comparar as informações similares para um dado som ou que tenham mais ou menos uma determinada propriedade. Para permitir esta busca o espaço vetorial criado deve satisfazer a seguinte restrição para cada propriedade a ser utilizada na recuperação do dados. Primeiramente os sons divergentes em relação a uma dada propriedade devem ser mapeados em diferentes regiões do N-espaço para permitir que o banco de dados possa fazer distinção entre diferentes valores desta propriedade. Além disso deve garantir que todas as propriedades relevantes tenham sido consideradas na construção do vetor para atender a um número máximo de propriedades que o usuário possa especificar na consulta. Os diversos aspectos do som como altura, sonoridade, brilho e harmonicidade podem variar com o tempo e sua trajetória deve ser calculada e armazenada em função de parâmetros como: média, desvio padrão, autocorrelação, máximos e mínimos, pontos críticos, tempo de ataque, etc. Um vetor deve ser utilizado para medir a duração e os parâmetros citados para cada um dos aspectos do som (altura, harmonicidade, intensidade).

4 - Classificação de Dados Sonoros

Bancos de dados com recursos multimídia armazenam gravações de aúdio freqüentemente indexadas de forma muito pobre, no sentido de utilização de classificação. Geralmente a indexação é realizada a partir de atributos textuais aos quais o dado sonoro se relaciona, acarretando uma série de dificuldades no momento da busca. Uma delas é a ambigüidade na classificação do dado. Mesmo que o mantenedor do recurso tenha associado palavras chaves para classificá-lo, esta associação é bastante subjetiva e pode não ser compreensível para outra pessoa. A busca por um som particular ou uma classe de sons requer um processo de classificação que permita identificar dados sonoros a partir de seus atributos próprios, permitindo uma comparação entre eles tal que sons similares segundo uma determinada propriedade estejam mapeados numa mesma região do espaço de classificação. Uma vez que se tenha conseguido armazenar os sons numa estrutura adequada ao tipo de classificação que se pretende (Blum 1996B), como por exemplo, um Vetor_N, pode-se utilizar as propriedades definidas (altura, intensidade, duração, etc) para realizar a classificação de dados sonoros. Para cada novo som inserido no banco de dados, o vetor é calculado e quando ele precisa ser classificado, este processo é realizado a partir do cálculo da distância deste em relação ao modelo existente, levando em consideração o peso de cada característica para a classe tratada. A distância é comparada para determinar se o som está dentro ou fora da classe. Se ele se enquadrar em mais de uma classe, ele será colocado naquela da qual ele for menos distante. Para bancos de dados de pequeno volume torna-se fácil calcular a distância para todos os sons e depois procurar aqueles que atendem a um certo resultado. Para aumentar a velocidade de busca em banco de dados muito grandes, pode-se indexar os sons por todas as características acústicas.

5 - Consultas por Similaridade Melódica

A consulta por similaridade melódica pode atender vários tipos de necessidade. Para responder a um usuário que precise extrair as obras que contenham uma melodia exatamente igual a da consulta, o sistema de busca deverá selecionar trechos musicais construídos a partir de uma sucessão de notas da mesma altura que a da melodia informada. Este tipo de consulta, pode ser feito inclusive a partir da escrita convencional de partitura, como na representação e recuperação baseada em conteúdo de partituras musicais em bases de dados orientadas a objetos proposta por M. Figueiredo no IV SBCM/1997. Contudo a extração da informação a partir da partitura convencional limita o universo de consulta, pois muitas vezes o usuário não contém conhecimento sobre escrita musical e mesmo para aqueles que o possuem, o interesse na recuperação de obras musicais a partir de similaridade melódica requer uma flexibilidade quanto à igualdade de freqüências. Um segundo grupo de usuários seria aquele interessado em formular consultas num banco de dados de música para obter obras cuja melodia fosse similar à informada, porém escritas num certo intervalo acima ou abaixo do original.

Consulta	Freqüência(Hz)	Nota recuperada	freqüência(Hz)	Intervalo
Do 3	130,81	Sol 3	196,00	(do3 a sol3) - 5a. justa
Sol 2	98,00	Ré 3	146,83	(sol2 a re3) - 5a. justa
Lá 2	110,00	Mi 3	164,81	(lá2 a mi3) - 5a. justa
Do 4	261,63	Sol 4	392,00	(dó4 a sol4) - 5a. justa

Tabela 1 – Relação de freqüências e intervalos de notas de um trecho de uma consulta e da obra recuperada, a partir de similaridade melódica

Desta forma, se o usuário informasse um trecho melódico contendo as notas do, sol , la e do, músicas com uma melodia que contivessem estas notas exatamente ou por exemplo, sol, re, mi e sol deveriam ser recuperadas. No exemplo dado, um intervalo de quinta justa é mantido entre cada das quatro notas da melodia. Ainda que a freqüência de cada nota comparada seja diferente, a diferença de altura permanece a mesma.

Uma terceira classe de consulta por similaridade melódica seria aquela de aproximação das alturas. Neste caso, tomaria-se como base a primeira nota da seqüência e a partir dela as demais seriam classificadas segundo a sua altura em relação à nota imediatamente anterior. Para esse fim a escala de sons pode ser reduzida a um alfabeto restrito que indique a altura relativa de cada nota em relação à anterior. Considere como exemplo, a conversão da introdução do tema da Quinta Sinfonia de Beethoven na seguinte seqüência: - B B A C B B A. A primeira nota é ignorada por não possuir uma anterior na seqüência que permita a comparação. Excetuando a primeira nota, à cada uma das demais da seqüência é atribuída uma das letras A, B, ou C conforme a altura da nota seja inferior, idêntica ou superior à anterior. Se utilizarmos um alfabeto de três letras para indicar as alturas das notas da seqüência dada no exemplo de consulta anterior(do3, sol2, la2, do4) chegaremos à seguinte seqüência: (-, A, C, C). Utilizando este método para representar um trecho da melodia de uma música, o sistema para recuperação de informação musical em banco de dados de áudio proposto por (Ghias 1995) mostrou-se capaz de recuperar muitas melodias de doze notas dentre as 183 armazenadas. Para um volume pequeno de músicas a ser pesquisado, este método pode trazer a vantagem da rapidez, em detrimento da precisão. Contudo em banco de dados de grandes volumes de melodias, uma resolução maior seria necessária. Para isso o alfabeto inicialmente considerado deveria ser expandindo. Outras pesquisas em técnicas para reconhecimento de padrão melódico como em (Crawford 1997) sugerem o uso de técnicas empregadas no reconhecimento de padrão de cadeias de caracteres, devido a similaridade entre a análise da música tradicional e os problemas computacionais de reconhecimento de seqüências de caracteres. O músico interessado em análise musical está freqüentemente preocupado em encontrar ocorrências de padrões ou repetições de um mesmo padrão, possivelmente com variações, da mesma forma que os algoritmos de computador têm que executar tarefas similares sobre cadeias de caracteres, as quais são seqüências de símbolos de algum alfabeto. Muitos objetos podem ser considerados como seqüências de caracteres: um arquivo texto ou também uma partitura musical. Ela pode ser vista como uma seqüência de símbolos de um alfabeto de notas. Desta

forma podemos trabalhar com estruturas polifônicas, formulando os seguintes tipos de consultas:

i.Dado um conjunto de seqüência de notas(uma para cada voz), descubra se um subseqüência exata ocorre em uma das vozes.

ii.Dado um conjunto de seqüência de notas(uma para cada voz), descubra se uma subseqüência exata ocorre em uma das vozes sem preservar o tempo de duração de cada padrão.

iii. Dado um conjunto de seqüência de notas(uma para cada voz), identifique padrões de repetição em vozes diferente ou repetidos na mesma voz.

iv. Dado um conjunto de seqüência de notas(uma para cada voz), e um padrão, descubra se ele ocorre em uma das seqüências dadas tanto na forma original, invertida, retrógrada ou com inversão retrógrada.

Utilizando o conhecimento rítmico da música pode-se apurar o reconhecimento de melodias similares, acrescentando à informação de altura, o acento característico do tempo em que a nota é ouvida. Segundo (Bakhmutova 1997) quase sempre fragmentos similares de uma música podem ser descobertos em outras a partir das chamadas repetições de segundo tipo ou adaptações. Através da representação de intervalos entre notas, um trecho musical pode ser representado como uma seqüência do tipo IkSk, K = 1,2, ..., N - 1, onde Ik é o intervalo de alturas entre a K-ésima nota e a K+1 e Sk representa a acentuação métrica dos sons. Esta acentuação é caracterizada pela divisão em compassos, que distribui a melodia em tempos fortes e fracos. Nesta representação tem-se a possibilidade de comparar duas melodias e extrair padrões de repetições não somente na variação de altura, mas também no elemento rítmico, que ajuda a definir melhor a melodia. Utilizando este método para comparar as duas seguintes seqüência de notas N2 e N3 à seqüência original N1, num compasso 6/8, teremos:

N1 : (nota)	do3,	sol2,	la2,	sol3,	la2,	sol2, do3
(tempo)	1,2, 3		1,2, 3,		4,5, 6	1,2,3,4,5,6
(acentuação)	(F)		(F)		(F)	(F)
N2 : (nota)	sol3,	re3,	mi3,	re4,	mi3,	re3, sol3
(tempo)	1,2, 3		1,2, 3,		4,5, 6	1,2,3,4,5,6
(acentuação)	(F) -		(F) -		(F) -	(F)
N3 : (nota)	sol3,	re3,	mi3,	re4,	mi3,	re3, sol3
(tempo)	1,2, 3		1,2, 3,		4,5, 6	1,2, 3,4,5,6
(acentuação)	(F) -		(F) -		(F) -	(F) -

N1 : (3-+) (1+-) (6++) (6--) (1-+) (3+ -)
N2 : (3-+) (1+-) (6++) (6--) (1-+) (3+ -)
N3 : (3-+) (1+-) (6++) (6--) (1- -) (3++)

A representação (3-+) presente no início dos três trechos musicais N1, N2 e N3, significa que a distância entre a primeira nota desta passagem(do3) e a próxima(sol2) equivale a três notas(si,la,sol). O sinal negativo indica a direção do intervalo, que no caso passa do mais agudo(do3) para o mais grave(sol2). O sinal positivo indica que a acentuação forte do compasso não recai sobre a última nota(sol2) do intervalo representado. No compasso de seis tempos utilizado no exemplo acima, o próximo tempo forte do trecho N1 cairá sobre a terceira nota(la2). Por esse motivo a representação do intervalo(sol2-la2) terá o sinal de acentuação negativo(1+-).

Neste tipo de representação mesmo utilizando-se a mesma seqüência de notas com variação apenas na distribuição rítmica, as seqüências N1 e N3 são consideradas apenas parcialmente similares, pois os dois últimos intervalos de N3 não apresentam semelhança rítmica com os de N1.

6 - Consultas por similaridade de atributos sonoros

Para usuários que estejam interessados em outros atributos da música diferentes da melodia, como compositores preocupados com aspectos timbrísticos do som, por exemplo, o sistema deve permitir a formulação de consultas por valores, levando em consideração as propriedades desejadas. Utilizando um vetor para armazenar as propriedades desejadas da consulta, pode-se excluir aquelas que o usuário considere desprezíveis em termos da sua finalidade e montar tipos de consultas para recuperação de sons a partir das suas propriedades.

- Consulta por Valor: Este tipo de consulta deve ser capaz de recuperar todos os sons cujos valores de uma certa propriedade Pi sejam maior, igual ou menor que um determinado valor.

Ex.: Recupere todos os sons cujos valores da propriedade P0 >= 0.5 e propriedade P1 < 0.3

- Consulta por exemplos: Este tipo de consulta deve poder recuperar todos os sons similares a uma certa amostra, em relação à propriedade p0. A similaridade neste caso pode ser obtida recuperando todos os sons dentro de uma faixa de valores para a propriedade p0.

Ex.: Recupere todos os sons cuja propriedade P0 >= 0.3 e P0 < 0.5

7 - Conclusão

Empregando formas de classificação que utilizam pesos diferenciados aos atributos da seleção, se soubermos a priori que algumas características acústicas não são importantes para a classe em questão, podemos ignorá-las ou atribuir-lhes um peso pequeno no cálculo da distância. Utilizando métodos que não dependem das características particulares de áudio e que podem ser aplicados à recuperação de imagem, como o proposto por (Foote 1997), consegue-se um bom resultado quanto à classificação de dados sonoros em relação à sua altura. Contudo a classificação segundo uma determinada característica, por exemplo, o

timbre, fica prejudicada, uma vez que este método não utiliza a atribuição de pesos a cada uma das propriedades tratadas. Na classificação de um som de oboé, por exemplo, este método tende a aproximar sons de alturas similares, podendo colocar na mesma classe outros instrumentos como trombone ou violoncelo, baseando-se na altura. Outra linha de pesquisa (Feiten 1994) tem abordado o problema da classificação em banco de dados de áudio utilizando redes neurais. Contudo, esta prática ainda apresenta alguns problemas devido ao fato de ser difícil olhar dentro da rede depois que ela é treinada ou durante a operação para determinar como o treinamento ocorre ou quais aspectos dos dados sonoros são similares a outros. Desta forma, torna-se difícil ao usuário especificar quais as características do som são importantes e quais aquelas que podem ser ignoradas para fins de classificação.

8 - Agradecimentos

Esta pesquisa vem sendo apoiada pelo CNPQ e pela Escola de Música da UFRJ. Meu especial agradecimento ao professor Rodolfo Caesar por seus valiosos comentários.

9 - Referências

Bakhmutova, I.; Gusev, V. & Titkova, T. (1997). The Search for Adaptations in Song Melodies. *Computer Music Journal,* 21:1.

Blum, Thom; Keislear, Doug; Wheaton, James & Wold, Erling. (1996). Audio Analysis for Content-Based Retrieval. *Tech. rep., Muscle Fish LLC, 2550 Ninth St., Berkeley, CA, USA.*

Blum,Thom; Keislear, Doug; Wheaton, James & Wold, Erling (1996). Content-based classification, search and retrieval of audio. *IEEE Multimedia..*

Crawford, Tim; Iliopoulos, C. S & Raman, Rajeev. (1998). String Matching techniques for Musical Similarity and Melodic Recognition. *http://www.kcl.ac.uk/kis/schools/hums/music/ttc/String_matching.html*

Feiten, Bernard & Günzel, Stefan (1994). Automatic Indexing of a Sound Database Using Self-organizing Neural Nets. *Computer Music Journal,* 18:3.

Foote, Jonathan. (1997). Content-Based Retrieval of Music and Audio, *Multimedia Storage and Archiving System II*, Proc. of SPIE, Vol 3229, pp 138-147.

Ghias, A.; Logan, J.; Chamberlin, D. & Smith, B. (1995), Query by Humming - Musical information retrieval in an audio database. *ACM Multimedia*, Electronic Proceedings, San Francisco, Califórnia, 1995.

Hourdin, Christophe; Charbonneau, Gérard & Moussa Tarek (1997). A Multidimensional Scaling Analysis of Musical Instruments's Time-Varying Spectra. *Computer Music Journal,* 21:2.

Opren, Keith & Huron, David (1991). The Measurement of similarity in music: A quantitative approach for non-parametric representations. *Computers in Music Research*, n. 4.

Lindsay, Adam (1998). Using Contour as a Mid-Level Representation of Melody. *S. M. Thesis*, MIT Media Laboratory, Cambridge, MA.

McNab, Rodger; Smith, Lloyd; Bainbridge, B. & Witten, Ian (1997). The new Zealand Digital Library Melody Index. *In D-Lib Magazine.*

Pierce, J. R. (1984). *Le Son Musical.* Editions Pour la Science - Paris.

Schaeffer, Pierre & Guy Reibel. *Solfège de l'objet sonore.* Production du Groupe de Recherches Musicales de l'O.R.T.F., Editions du Seuil.

Vercoe, Barry L. (1992). *Csound - Manual for the Audio Processing System and Supporting Programs.* M.I.T., Cambridge, Massachusetts.

Mind the Music:
Towards Thought-Controlled Music Systems

ALEXANDER DUNCAN
EDUARDO RECK MIRANDA
KEN SHARMAN
Centre for Music Technology - University of Glasgow
14 University Gardens, Glasgow, G12 8QQ
Scotland, United Kingdom
{aduncan, eduardo, ken}@music.gla.ac.uk

Abstract

We are investigating whether electroencephalogram (EEG) patterns can be detected and classified according to different auditory stimuli. We explore advanced multichannel signal processing techniques to perform pattern classification on the EEG data. This paper outlines our research methods and describes the ongoing experimental work which uses a 128-channel dense array EEG recording system. This is the first step in our attempt to explore ways of allowing people to control a musical environment with their mind.

1. Introduction

The ultimate goal of this research project is to develop a system whereby people can create music simply by thinking. It is too much of a leap in the dark to foresee the particulars of such a system at this stage, but it is not inconceivable to imagine a situation in which one could actively listen to a type of musical composition that actually changes its course, rhythm, mood, style, etc. according to the brainwave signals of the listener. An even more ambitious goal would be a system that could actually synthesise the sounds (or music, for that matter) one is thinking of (Figure 1).

We envisage a system in which a wearable bio-feedback device, furnished with a neuro-compatible interface, would produce some sort of raw 'musical signal' that gets shaped (or 'composed') by the mind of the listener. Specific types of compositions could then be devised for particular cases; for example, for a music therapy context, where the music is created in response to particular brain signals that causes either or both, relief of stress or states of arousal. This is a dream waiting to be realised, and the inspiration behind our research project, in which we believe a great number of benefits exist such as: making music creation accessible to a broader population (including people with impaired muscular abilities), adding to the understanding of music and mind, and promoting the development

of thought-controlled systems. The medical sciences will also benefit, as the quantitative analysis of the EEG is becoming more common in the research of medical conditions.

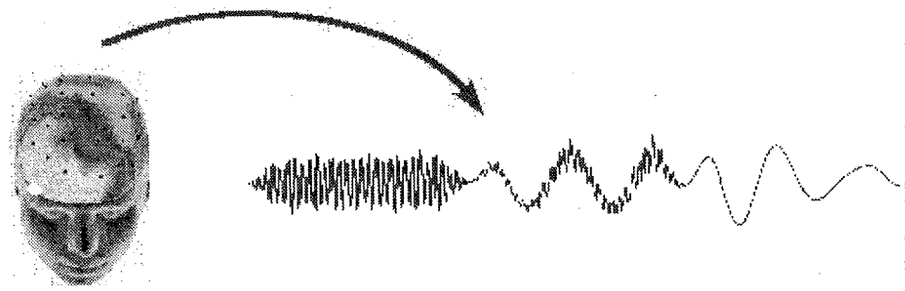


Figure 1: Imagine if it were possible to play music simply by thinking.

One of the key features that distinguish humans from other animals is the fact that we are intrinsically musical. Music is generally associated with the expression of emotions [Harrer and Harrer, 1977], but it is also common sense that the intellect plays an important role in musical activities [Deutsch, 1977]. All the same, music requires the ability to recognise and imagine patterns of sounds; it requires sophisticated memory mechanisms, involving both conscious manipulation of concepts and subconscious access to millions of networked neurological bonds [Miranda, 1997].

Our understanding of the behaviour of the brain when we engage in any type of musical activity (e.g. playing an instrument or simply imagining a melody) is merely the tip of an iceberg. A variety of methods to measure the activity of the brain have brought to light important issues that have helped researchers uncover the tip of the iceberg. The electroencephalogram method (EEG), for example, has been particularly useful to demonstrate that the brain expects sequences of stimuli that conform to established circumstances. For instance, if you hear the sentence "A musician composes the music", the electrical activity of your brain will tend to run fairly steadily. But if you hear the sentence "A musician composes the dog", the activity of your brain will display significant negative electrical response immediately after the word "dog". The human brain seems to respond similarly to musical events. This aspect of the brain activity is one of the main motivations of our research.

2 Measurements of brain activity

"The human brain produces a complex, multidimensional, pulsating, electromagnetic field resulting from the electrochemical behaviour of masses of neurones acting in small to very

large groups" [Rosenboom, 1990]. The EEG is the mixed frequency electrical signal produced by this activity, and it is measured from electrodes placed on the scalp. We found a few encouraging references in both musical and non-musical research fields that confirm that the EEG provides a rich source of information about our thought (musical) processes: [Birbaumer, 1994], [Janata, 1994], [Makeig, 1997] and [Saiwaki, 1997].

EEG data have been categorised into four main components [Rosenboom, 1990]:

(a) a random-seeming background signal
(b) long-term coherent waves
(c) short-term transient waves
(d) complex ongoing waves

The random-seeming background signal is the residue observed after all known methods of waveform decomposition are exhausted; very little is known about this signal. Long-term coherent waves are the well-known *alpha, beta, delta,* and *theta* rhythms, which range from approximately 1 to 30 Hertz. They are often associated with certain states of consciousness, such as alertness and sleep. Short-term transient waves reflect the 'singular experience' associated with an external stimulus and up to now they have been accessible only by Event Related Potential (ERP) analysis. ERPs are derived by taking the average of many EEG recordings, where the person is subjected to the same stimulus repeatedly. The reason for averaging is to remove noise caused by other uncorrelated brainwave activities. ERP analysis has played an important part in developing the theory of music cognition, as authors like Rosenboom [Rosenboom, 1990] and others have shown. Finally, it is suggested that a non-random complex component exist, whose ever changing pattern comes from the build up of baseline activation's from the vast neuronal masses within the brain. This pattern is expected to be the result of the ongoing, self-organisation of information during a person's own life experience. If these patterns could be successfully measured, and sense made of them, we might witness the mechanisms of higher level thought processes.

Another measurement of brain activity is the magnetoencephalogram (MEG), which uses super-conducting sensors to measure the fluctuation and topographic distribution of the magnetic fields associated with the discharge of neural action potentials, and electrochemical activity within the brain. The advantages of MEG over the EEG are twofold: (a) signals associated with highly spatial-localised activity are more readily detected; and (b) the sensing equipment does not require direct contact with the head, thus making it more comfortable. MEG technology is, however, still very expensive

Other methods of measuring brain activity, such as Magnetic Resonance Imaging (MRI), and Positron Emission Tomography (PET), provide a mapping of the activation of individual cortical areas. They can contribute to the understanding of the spatiotemporal

activity of the brain under certain conditions; for example, PET scans have shown that listening to music and imagining listening to music 'activate' different parts of the brain. Unfortunately, both MRI and PET require extremely expensive and ergonomically restrictive equipment.

Mainly due to its temporal resolution, EEG technology is currently favourable for methods that might lead to 'expert-thought-systems', that is, those mind-controlled.

3 An overview of the ongoing research work

At present we are developing technology to detect features of brain activity that represent the sounds one thinks of. However, it seems logical that a primary aim might be to isolate general sonic-thought features, from the other ongoing mental activities; i.e. to extract information about musical thoughts from the brainwaves. We have chosen, at least for now, to focus on exploring methods of EEG analysis. We are also investigating methods to map brainwave signals to the parameters of a computer sound synthesis system [Miranda, 1998].

We currently are investigating whether spatiotemporal frequency patterning could be achieved by EEG analysis using a combination of auto-correlation (to segment EEG time series data), auto-regression (as an alternative to Fourier methods), and directed coherence analysis techniques, which form a type of statistical information flow method. Work of this nature has shown that there is a relationship between EEG and music cognition [Saiwaki, 1997] but these techniques still need to be much refined and adapted to suit our purposes.

Statistical time-domain methods such as, Independent Component Analysis (ICA) and Principal Components Analysis (PCA) are currently being used to look for evidence of voluntary musical thought in EEG. ICA, for example, has already been successfully used to segment auditory-stimulated ERPs into separate components [Makeig, 1997]. The strengths of these components were found to be related to specific features of the human cognitive process, involved in the detection of specific aspects of auditory stimulation. ICA can separate N statistically separate inputs, which have been mixed linearly at N outputs, therefore, the number of components equals the number of electrodes. We are currently assessing whether ICA can be effectively used as a pre-processing tool for further statistical pattern recognition methods, or as inputs to artificial neural networks (ANNs) [Hertz et al., 1991].

We believe that the application of ANNs for extracting musical thought features from pre-processed EEG signals might be the best approach to the problem (Figure 2), as they have been successfully applied in other similar problems; e.g. for the recognition of epileptic seizures [Weng and Khorasani, 1996].

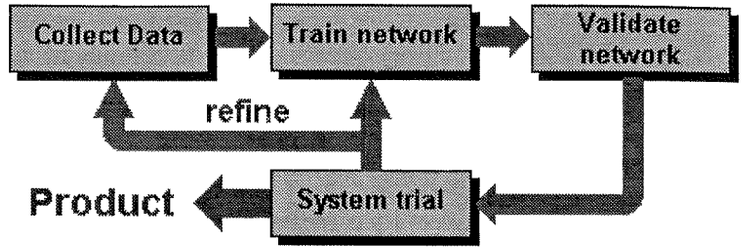


Figure 2: *An artificial neural network system, to 'represent' the brain activity generated when one thinks of sounds, is under development*

4 Measuring musical thoughts: our approach to the problem

We are running a number of experiments to test the hypothesis that we can distinguish between the EEG of a subject when s/he is hearing a simple musical sound or not, with a view to further experiments that incorporate imagined musical events as well.

A group of subjects are presented with an auditory stimulus. During the experiment, the subjects listen to a short audio track, presented through loudspeakers. The audio track consists of a repeated recording of a sound, spaced by random moments of silence. Each subject is presented with the same audio track several times.

The EEG equipment records a synchronisation pulse generated by the stimulus device to ensure that the EEG data is 'marked' according to the exact time the sounds are presented.

Most of the work to date that involves EEG analysis use the time-averaged Event Related Potential technique (ERP). However, we adopted a different approach: we treat the data set as a multichannel time series, and make as few a priori assumptions as possible.

The five main stages of our experiments are discussed below.

4.1 Bio-sensing

The bio-signal we are measuring comes from the EEG equipment and the data that are collected form a multichannel time series. This time series can be thought of as a multidimensional pattern space.

The EEG equipment we are using is the state-of-the-art 128-channel *Geodesics Sensor Net*, by Electrical Geodesics. The Geodesic Sensor Net is composed of 128 plastic sensor housings, interconnected by durable polyurethane elastomer threads that form the tension

lines of an icosahedron. Each sensor housing contains a low drift, custom-made electrode embedded in an electrolytic sponge. As the sensor net is stretched over the head, the electrodes make contact with the scalp via the sponges and are held firmly by the geodesic tension network (Figure 3).

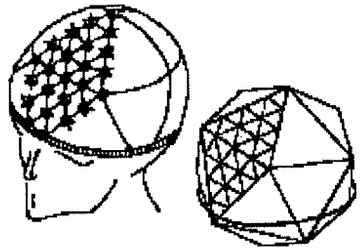


Figure 3: With a 2.8 cm intersensor distance, the Geodesic Sensor Net provides adequate means for the characterisation of the electrical fields on the scalp.

4.2 Pattern localisation and normalisation

Once the raw EEG data has been collected, some initial pre-processing is required to remove unwanted 'contamination', such as muscular and eye movement artefacts. A simple amplitude threshold test can achieve this [Jung et al., 1997]. At this stage, the data also require additional pre-processing operations, such as normalisation and scaling, zero padding and windowing.

4.3 Feature extraction and selection

Both non-transformed signal characteristics, such as parametric modelling, and transformed characteristics, such as auto-regression and Fourier analysis, are options for the extraction of features from the pattern space. Here, statistical reasoning plays an important role in the selection of the best features for classification purposes.

4.4 Property formation and clustering

Statistical techniques, such as Principal Components Analysis (PCA), Independent Component Analysis (ICA), distance classifiers and clustering, combined with connectionist methods such as Kohonen networks (Figure 4), are among the methods we are investigating for the selection and property formation stages of the problem.

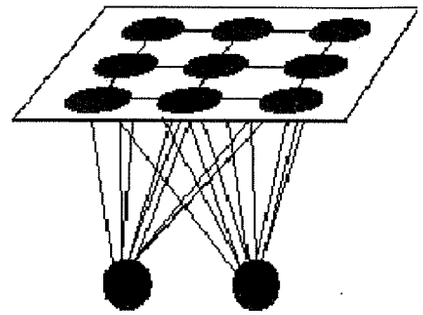


Figure 4: A Kohonen network is able to classify data by grouping them into clusters containing similar features.

4.5 Classification

For classification methods, we are investigating both statistical and connectionist (i.e. artificial neural networks) techniques, as both have shown to be effective in work by other authors. For example, neural networks have done successful detection of epileptic seizure patterns in EEG, where the network was trained with a known seizure EEG data-set [Weng and Khorasani, 1996]. Radial Basis Functions (RBF) is one of the techniques under investigation. RBF is a supervised feedforward neural network that combines linear and non-linear functions. RBF is suited for classifying data highly corrupted with noise.

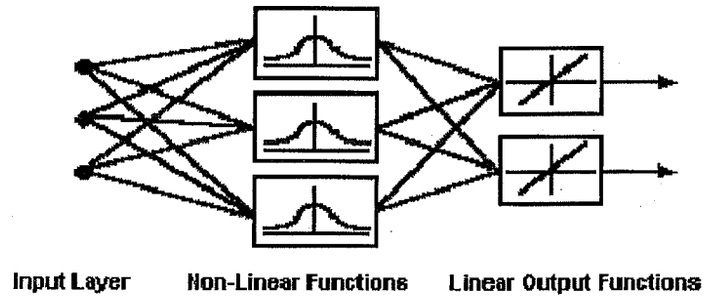


Figure 5: Radial Basis Functions is a supervised feedforward neural network that combines non-linear and linear functions.

5 Conclusion

The work outlined in this paper is the first step towards our long term goal of developing thought-controlled musical systems. It is our hope that sufficient information representing thought processes are present in the EEG, otherwise we may be searching for the impossible.

So far, we have learned that the design of suitable experiments is far from trivial, and possibly controversial. However, we have attempted to keep our first experiments simple and rigorous, with the hope that they lead to confidence and experience in handling multichannel time series EEG data. There are still many issues to explore, such as choosing the best analysis techniques, and developing the means to evaluate their success.

The EEG signals are very small and are immersed in an ocean of background noise. Also, the EEG signals appear to be variable between different subjects. We believe that analysis methods used in radio astronomy and sonar systems may help to provide clues as to the analysis methods we investigate; these systems also deal with highly noisy signals.

References

[Birbaumer, 1994] Birbaumer, N., "Perception of Music and Dimensional Complexity of Brain Activity", www.ccsr.uiuc.edu/~gmk/Papers/MusicEEG/ChaosMuTR2.html.

[Deutch, 1977] Deutsch, D., "Memory and Attention in Music", Music and the Brain, *Critchley M., Henson R.A. (Eds), London (UK): Willian Heinemann Medical Books Limited.*

[Harrer and Harrer, 1977] Harrer, G. and Harrer, H., "Music, Emotion and Autonomic Function", Music and the Brain, *Critchley M., Henson R. A. (Eds), London (UK): Willian Heinemann Medical Books Limited.*

[Hertz et al., 1991] Hertz, J., Krogh, A. and Palmer, R. G., Introduction to the Theory of Neural Computation, *Wokingham (UK): Addison-Wesley.*

[Janata 1994] Janata, P., "Probing Human Cognition with Music and Measures of Brain Electrical Activity", http://caja1.uoregon.edu/~janata/pubs/ks94_janata_html/ks94_janata_title.html.

[Jung et al., 1997] Jung, T., Makeig, S. and Stensmo, M., "Estimating alertness from the EEG Power Spectrum", IEEE Transactions on Biomedical Engineering, *Vol. 44, pp. 60-69.*

[Rosenboom, 1990] Rosenboom, D., "Extended Musical Interface with the Human Nervous System", Journal of the International Society for the Arts, Sciences and Technology, *Leonardo Monograph No.1.*

[Makeig, 1997] Makeig, "Blind Separation of Auditory Event-related Brain Responses into Independent Components", Proceedings of the National Academy of Science, *Vol. 94, pp. 10979-10984.*

[Miranda, 1998] Miranda, E. R., Computer Sound Synthesis for the Electronic Musician, *Oxford (UK): Focal Press.*

[Miranda, 1997] Miranda, E. R., "La música, las máquinas, la inteligencia y el cerebro", Letra Internacional *(Spain), No. 53, pp. 40-43.*

[Saiwaki, 1997] Saiwaki, N., "An Approach to Analysis of EEG Recorded During Music Listening.", Journal of New Music Research, *Vol.26, pp. 227-243.*

[Weng and Khorasani, 1996] Weng, W. and Khorasani, K., "An Adaptive Structure Neural Networks with Application to EEG Automatic Seizure Detection.", Neural Networks, *Vol. 9, No. 7, pp. 1223-1240.*

Interactive Control of Musical Structures by Hand Gestures

PAUL MODLER
Staatliches Institut für Musikforschung PK
Tiergartenstraße 1, D-10785 Berlin, Germany
pmodler@compuserve.com

1 Abstract

This paper summarizes our ongoing work on the mapping of hand gestures to musical parameters in an interactive music performance and virtual reality environment.
We use data obtained from a sensor glove, a high end input device for digitizing hand and finger motions into multi-parametric data. These data are processed in order to extract meaningful data to control musical structures.
For that we use a neural network architecture to achieve meaningful control parameters.
We focus on the mapping of gestural variations onto equivalent musical parameters, motivated by a creative situation like a performance. We set up a dictionary of symbolic and parametric subgestures. Different hand gestures of this dictionary and characteristic variations will be evaluated with respect to their applicability to intuitive and musical control of musical structures.
The system is complemented with a 3D VRML environment, i.e. an animated hand model and behaving representations of musical structures. This 3D representation combines with the gesture processing module and the sound generation to powerful so called Behaving Virtual Musical Objects.

2 Components of the System

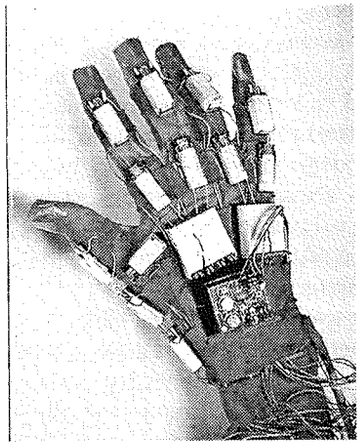
The interactive system comprises the following components which are described below in greater detail.

- a dedicated sensor glove which tracks hand and finger motions

- a design and control environment for the data glove including preprocessing features (written in JAVA)
- a data processing section based on neural networks for gesture recognition and postprocessing
- a real-time sound synthesis module for advanced synthesis algorithms
- a virtual reality framework (VRML) for the interaction of the performer with virtual objects whichis included in the JAVA environment

3 Digitization of Hand Movements

The sensor glove developed by Frank Hofmann is used as an input device (Picture 1).



Picture 1 Sensor Glove Version 3 (by Frank Hofman)

By tracking 24 finger ankles and 3 hand acceleration values, gestures of the hand can be processed by the connected system. As a first approach, direct mapping of single sensor values to sound parameters was used. Although good results concerning the possibilities of controlling parameters of the sound algorithm (Frequency Modulation, granular Synthesis, analog Synthesis) have been obtained, disadvantages of such a direct connection occurred as well. E.g., intuitive control of multiple parameters simultaneously turned out to be hard to realize. The data from the Sensor Glove are fed into a postprocessing unit which provides feature extraction and gesture recognition abilities, as well as agent features for intelligent control of the subsequent sound synthesis module.

4 Separation of Gestures

We assume that a gesture consists of subgestures of symbolic and parametric nature (cf. Modler & Zannos 1997). The symbolic type does not have to be time-invariant. It can as well be a time-varying gesture to which the user denoted symbolic contents.
The parametric type should always be time-variant for the control of sound parameters.
With this subgesture architecture gesture-sequences can be recognized using only a part of the hand as a significant symbolic sign, while other parts of the hand movement are used as a parametric sign.
Subgestures allow for both, the application of smaller neural nets as well as the possibility of using trained nets (subgestures) in various compound gestures.
We aim at setting up a set of gestures, suited for the gestural control of musical parameters. The gestures are subdivided into symbolic and parametric subgestures as described above. We show how a dedicated neural network architecture extracts time varying gestures (symbolic gestures). Besides, secondary features such as trajectories of certain sensors or overall hand velocity will be used to extract parametric values (parametric gestures).
Special attention is given to the way a certain gesture can be emphasized or altered with significance for both emotions and music. We are investigating whether the concept of symbolic and parametric gestures can adequately describe the situation of an emotionally expressive performance.
The set of gestures will be evaluated regarding their potential of providing meaningful symbolic and parametric subgestures, as well as how these subgestures can deal with gestural variations.

4.1 Dictionary of Symbolic Subgestures

A set of 15 to 25 gestures was selected and used as a prototype dictionary. For a preliminary classification the following categories were used to organize the gesture dictionary.

A) gestures with short (not relevant) start and end transition phases and on static state (pose) (e.g. finger signs for numbers)
B) gestures with repetitive motions (e.g. flat hand moving up and down [slower])
C) simple (most fingers behave similar) gestures with relevant start and end state and one transition phase (e.g. opening hand from fist)
D) complex (most fingers behave differently) gestures with relevant or not relevant start and end states and transition phase
E) compound gestures with various states and continuos transitions.

The dictionary is based mainly on gestures of categories B) C) and A).
Since category A) contains poses, those types of instances have been selected as part of the dictionary, which the performer can use as very clear signs.
Few examples of category D) have been chosen, because of their more complex character.

4.2 Dictionary of Parametric Subgestures

Besides the symbolic gestures a set of parametric gestures has been selected in order to set up a dictionary for classification the subgestures according to the following categories:

a) alteration of the absolute position of the hand: translation
b) alteration of the absolute position of the hand: rotation
c) alteration of velocity (energy)
d) alteration of the cycle duration

In the dictionary of parametric subgestures we included instances of categories a), b) and c). Additional work has to be done regarding the extraction of repetitive cycle time and detection of resulting timing variations.
For the prototype implementation, an agent-type module which supervises the combination of symbolic and parametric subgestures has been included. This coordinating device recognizes the influence the extracted subgestures have on the output of the symbolic gestures.

5 Pattern Recognition of Gestures by Neural Networks

5.1 Neural Network Architecture

Based on a prototype implementation for the recognition of continuos performed time-variant gestures (cf. Modler & Zannos 1997) we extended the proposed architecture to deal with the demands of the selected dictionary. Additional input layers have been added for the recognition of subgroups of the gesture dictionary. The layers of the subgroups are connected by separate hidden layers.

5.2 Training Procedure

The Network is trained with a set of recorded instances of the gestures of the symbolic subgesture dictionary.
Both the 2D representation of the sensor data as well as the 3D-representation (section 6.2) offered a good feedback about recorded instances.
Each gesture class was recorded two times in 4 different velocities.
The training of the Neural Net was conducted offline. The resulting net parameters were transferred to the Sensor Glove processing section and integrated in the C/JAVA environment.

5.3 Recognition of Subgestures by Neural Networks

For evaluation, time-varying continuos connected phrases of instances of the symbolic subgesture dictionary were presented to the trained net. This was realized online, i.e. the data were passed directly from the glove to the network.

For the selected part of the gesture dictionary the proposed net architecture produced good results, i.e. a recognition rate of about 90 %.

5.4 Extraction of Parametric Subgestures, and Combination with Symbolic Subgestures

The parametric subgestures as proposed in section 5.2 were achieved by online processes. Further investigations will show whether neural networks can also provide the desired parameters.
The combination of both of them produced good results in both recognition of a subgesture and altering the overall gesture by altering the parametric subgesture (e.g. flat hand, fingers moving up and down [slower] combined with translation movements of the whole hand).

5.5 Results

The proposed combination of gesture recognition of symbolic subgestures with parametric ones brought up good results. In other words, they promise to become a powerful tool to extend the possibilities and variability of a performance conducted with the Sensor Glove.
The concept of symbolic and parametric subgestures as well as the proposed categories offer the performer a guideline to fix a parameter mapping with connected sound synthesis and visualization modules. The definition of Virtual Musical Objects (see below) is less cumbersome using this categorization.
The extension of the neural network for the processing of a larger number of features provided seems to be manageable, but an extension to a multiprocessing parallel architecture has to be considered, too.

6 3D Representation: Integration of Virtual Reality Worlds

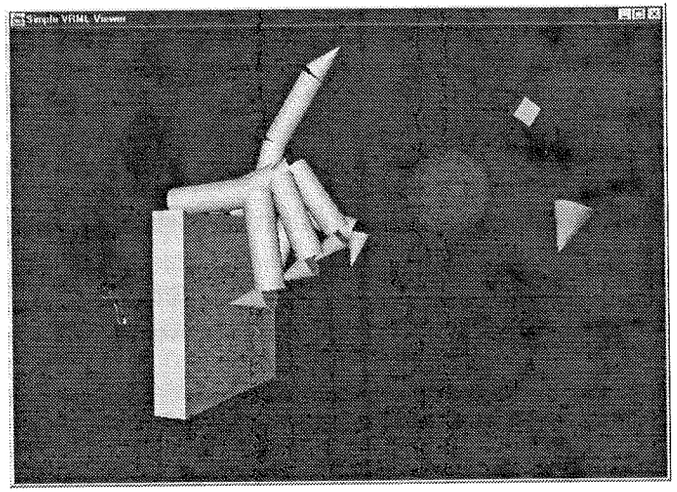
6.1 Animation of a Hand Model by the Sensor Glove

As a feasibility study, we have created a visual representation of a hand and Virtual Musical Objects in VRML language (cf. Picture 2).
The VRML language is a standardized toolkit which provides a potential for creating three-dimensional environments: virtual worlds. VRML offers the advantage of a platform and browser independent application. Since VRML is so widely accepted, its disadvantage of reduced speed is acceptable.
The hand model is animated with the input from the Sensor Glove. This is realized by a JAVA-VRML interface.
This prototype world can be viewed with a VRML browser that has been integrated into the design and control environment and runs on a combination of JAVA and C.
The VRML - JAVA interface also offers the possibility of dynamically creating or altering existing VRML worlds, in other words, user-provided interaction models such as the ani-

mated hand model can then be introduced into unknown worlds (e.g. scenarios downloaded from somewhere else).
Complex worlds can be generated with special tools like MAX3D or VREALM. which then can be animated, investigated, altered, and viewed with the VRML browser.

6.2 3D Representation of Behaving Virtual Musical Objects (BVMO)

In addition to the hand animation, we developed a framework for the creation of VMOs.
These objects – in connection with the Sensor Glove - constitute the gesture processing section and the sound synthesis module. An extended form (EVMI) of a Virtual Musical Instrument (VMI) has been proposed by Alex Mulder (Mulder 1994).
The VMOs are variable in color, size, form, and position in the surrounding world. Additional features can be defined and controlled, e.g. time-varying alterations of a certain aspect such as color or motion trajectory. This can be regarded as a behaving VMO (BVMO) or a resident.
The graphical representations (e.g. the hand model) are realized in VRML. For data passing the JAVA, the VRML interface can be used. Good results have been achieved for animating the hand model and altering BVMOs by user input from the Sensor Glove.



Picture 2 VRML World with Animated Hand Model and Virtual Musical Objects

7 Musical Application of the System

The synthesis module is designed in order to provide the ability of controlling different sound algorithms in real-time by continuos control parameters. The algorithms are realized in SuperCollider, a special sound synthesis language offering real-time synthesis possibilities as well as LISP-style list processing and a limited SMALLTALK like class/object structure.
Different synthesis techniques (Frequency Modulation, Physical Modeling and Granular Synthesis) are compared and described with respect to their controllability through the Sensor Glove and the proposed processing system.

8 Parallel Processing Aspects

Currently the system is based on a 200MHz MAC compatible 604PPC covering the data acquisition and processing from the Sensor Glove. It also realizes the gesture preprocessing recognition and postprocessing. The VRML 3D-environment is implemented on a 200 MHz PC which communicates with the SensorGlove machine by MIDI. The sound generation is realized on a 200 MHz MAC compatible 604PPC which is controlled by the Sensor Glove machine as well by MIDI. The training of a the neural network was executed on a Silicon Graphics Indigo2.
The MIDI connections were chosen because of their variability in combining the possible external tools as well as because of their easy handling. Due to the high data rates produced by the Sensor Glove, a more powerful way of transmission has to be considered. Ethernet TCP/IP or Firewire connections are possible solutions, since the C/JAVA environment can be connected with standard software tools over these connections.
For growing neural networks and additional processing of the control data, e.g. detection of beats or cyclic structures etc., a wider distribution of the computing tasks has to be considered.
Besides, the sound synthesis engine profits from a parallel architecture. Promising tests with a dedicated communication protocol for parallel distributed audio processes MIDAS have been conducted.

9 Conclusions

Based on our experiments we come to the following results and conclusions.
The subgestural concept for deriving symbolic and parametric gestures is a good approach for integrating gesture recognition into a performance situation. The neural network pattern recognition is combined with flexible and intuitive possibilities of altering material.
Specific control changes as well as intuitive overall changes can be achieved.
The proposed categories of subgestures offer the performer a comprehensive way to design the behavior of the sound generation. This provides a powerful alternative to the one to one mapping of single parameters.
The proposed dictionary of gestures provides the performer with an intuitive way for musical expressiveness and meaningful variations.
Behaving Virtual Musical Objects integrated in a virtual 3D world offer a promising way for novel visual representation of abstract sound generation algorithms. This includes specific control of a sound scene, but also facilitates memorizing and recalling of a sound scene and inspires the user to new combinations.

The combination of the proposed gesture mapping with the BVMOs constitutes a powerful environment not only for interactive performances, but also for the design of sounds and sound scenes.

10 References

Hofmann, F.G, Hommel, G.: Analyzing Human Gestural Motions Using Acceleration Sensors., Proc. of the Gesture Workshop '96 (GW'96), University of York, UK, im Druck

Hommel, G., Hofmann, F., Henz, J.: The TU Berlin High-Precision Sensor Glove. Proc. of the Fourth International, Scientific Conference , University of Milan, Milan/Italy, 1994

Modler, Paul, Interactive Computer-Music Systems and Concepts of Gestalt, Proceedings of the Joint International Conference of Musicology, Brügge 1996.

Modler, Paul, Zannos, Ioannis, Emotional Aspects of Gesture Recognition by Neural Networks, using dedicated Input Devices, in Antonio Camurri (ed.) Proc. of KANSEI The Technology of Emotion, AIMI International Workshop, Universita Genova, Genova 1997

Mulder, Axel, Virtual Musical Instruments: Accessing the Sound Synthesis Universe as a Performer, 1994
http://fassfu.ca/cs/people/ResearchStaff/amulder/personal/vmi/BSCM1.rev.html

Schmidt-Atzert, Lothar, Lehrbuch der Emotionspsychologie, Kohlhammer, Stuttgart Berlin Köln, 1996

SNNS, Stuttgarter Neural Network Simulator, User Manual 4.1, Stuttgart, University of Stuttgart, 1995.

Waibel, A., T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang, Phoneme recognition using time-delay neural networks, IEEE Transactions On Acoustics, Speech, and Signal Processing, Vol 37, No. 3, pp. 328-339, March 1989.

Wassermann P.D., Neural Computing, Theory and Practice, Van Nostrand Reinhold, 1993.

Wundt, W., Grundzüge der Physiologischen Psychologie, Verlag von Wilhelm Engelmann Leipzig, 1903

Zell, Andres, Simulation Neuronaler Netze, Bonn, Paris: Addison Wesley, 1994.

Sound Sculpting: Performing with Virtual Musical Instruments

AXEL G. E. MULDER
School of Kinesiology, Simon Fraser University, Burnaby, BC, Canada V5A 1S6.
Email amulder@sfu.ca

S. SIDNEY FELS
Dept. of Electrical and Computer Eng., University of British Columbia, Vancouver, BC, Canada V6T 1Z4

Abstract

We have prototyped an environment for designing and performing with virtual musical instruments. It enables a sound artist or musician to design a musical instrument according to his or her wishes with respect to gestural and musical constraints. Sounds can be created, edited or performed by changing parameters like position, orientation and shape of a virtual input device, that can only be perceived through its visual and acoustic representations. We implemented the environment by extending a realtime, visual programming language called Max/FTS, with software objects to interface datagloves and 6DOF sensors and to compute human movement and virtual object features. Our pilot studies involved a virtual input device with a behaviours of a rubber balloon and a rubber sheet for the control of sound spatialization and timbre parameters. It was found that the more physical analogies are used for a mapping the faster the mapping can be learned. We also found that we need to address more manipulation pragmatics in the mapping and more carefully identify movement features to map to virtual object parameters. While both hands can be used for manipulation, left-hand-only sound sculpting may be a useful replacement for and extension of the standard keyboard modulation wheel.

1 Introduction

We report in this paper on work in progress to develop gestural interfaces that allow for simultaneous multidimensional control. An example of simultaneous multidimensional control can be found in music composition and sound design, which task involves the manipulation of many inter-dependent parameters simultaneously. For any sound designer or composer and certainly a performer the control of these parameters

involves a significant amount of motor and cognitive processing to coordinate his or her motor system when mouse-and-keyboard interfaces are used. These interfaces capture only a very limited range of gestural expressions. Although the human hand is well-suited for multidimensional control due to its detailed articulation, the mouse and keyboard do not fully exploit this capability. In fact, most gestural interfaces, even those that use a dataglove-like interface do not fully exploit this capability due to a lack of understanding of the way humans produce their gestures as well as a lack of understanding what meaning can be inferred from these gestures [7].

Thus, to reduce the motor and cognitive load for the sound designer, it is necessary to design a gestural interface that implements data reduction with respect to the controlled parameters and/or an interface that exploits the capability of human gestures to effortlessly vary many degrees of freedom simultaneously at various levels of abstraction. In terms of data reduction of sound synthesis parameters, we use a sound synthesis environment that facilitates representation at various levels of abstraction of sound. In terms of exploiting human gestural capability, we use a virtual input device that senses and responds to manipulation by the entire hand and allows for a wide range of gestural expressions.

In addition, by creating intuitively related representations of the feedback from the gestural expression in various media, we hope the user's perceptual system will have more information about the control space and thus be able to decide faster between the different control possibilities. The general aim of this research is to create a multiple abstraction level, consistent and unified, yet user-adaptable input method for simultaneous multidimensional continuous control tasks such as sound design.

1.1 Sound Sculpting

We are evaluating the above approach by implementing sound sculpting [6] as a design environment for virtual musical instruments [8]. In sound sculpting, a virtual object is used as input device for the editing of sound - the sound artist literally "sculpts" sounds using a virtual sculpting computer interface [5], i.e. by changing virtual object parameters such as shape, position and orientation. In our study, the object is virtual, i.e. the object can only be perceived through its graphics display and acoustic representations, and has no tactile representation. Although sculpting in the physical world is most effective with touch and force feedback, our assumption is that these forms of feedback can be replaced by acoustic and visual feedback with some compromises. The motivation to make such assumptions is based on the fact that the generation of appropriate touch and force feedback, while exploiting the maximum gestural capability in terms of range of motion and dexterity, is currently technically

too challenging in a virtual object manipulation task.

1.2 Gestures

We are focusing on physical manipulation gestures, such as used for changing object shape (i.e. sculpting), in which the shape, position and orientation of the hand is changing simultaneously. Such gestures provide the highest dimensionality of control, especially when two hands are involved in the manipulation. In addition, if well designed, two-handed manipulation increases both the directness and degree of manipulation of the interface [2], thereby reducing the motor and cognitive load. We are also interested in the use of dynamic signs (hand shape is constant, but hand position and/or orientation is changing), although they represent less dimensions of simultaneous control. The other end of the spectrum, in terms of control dimensionality, is represented by static signs, which are not useful in the task we are interested in (i.e. multidimensional control) other than for selection tasks. In previous work on gesture interfaces such as [4] it has been noted that, since humans do not reproduce their gestures very precisely, natural gesture recognition is rarely sufficiently accurate due to classification errors and segmentation ambiguity. Only when gestures are produced according to well-defined formalisms, such as in sign language, will automatic recognition have acceptable precision and accuracy. However, a gesture formalism will require tedious learning by the user. Thus we do not compute or analyse any abstract symbolic representation of the gestures produced by the user, but instead focus on the continuous changes represented by the gestures.

1.3 Pragmatics

When we consider object manipulation as the changing of position, orientation and shape of an object, the pragmatics for position and orientation changes of small, light objects are simple and do not involve any tools. However, an analysis of the methods employed by humans to edit shape with their hands leads to the identification of three different stereotypical methods.

- **Claying** - The shape of objects made of material with low stiffness, like clay, is often changed by placing the object on a supporting surface and applying forces with the fingers of both hands.
- **Carving** - The shape of objects made of material with medium stiffness, like many wood materials, are often changed by holding the object in one hand and applying forces to the object using a tool like a knife or a file.

- **Chiseling** - The shape of objects made of material with high stiffness, like many stone materials, are often changed by placing the object on a supporting surface and applying forces to the object using a tool like a chisel held in one hand and a hammer held in the other.

2 Mapping Strategies

Sound sculpting should minimize the motor and cognitive load, or in other words be "easy to use", for a novice user of the system. The ease-of-use is determined by the design of the mapping. Figures 1 and 2 illustrate our approach of using virtual object parameters at various levels of abstraction as a means to relate hand movements to sound variations.

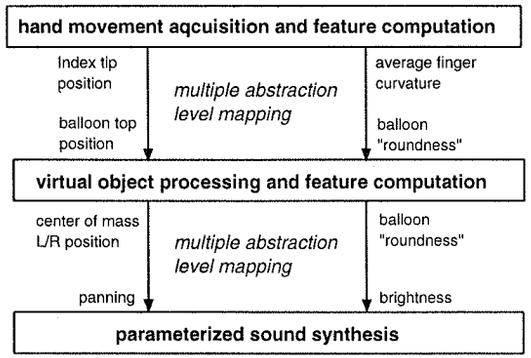


Figure 1: Functional diagram of hand movement to sound mapping. Virtual object features are used as a means to relate hand movement features to sound features. The diagram intends to illustrate representation-consistent mapping, i.e. the mapping of hand movement parameters to virtual object parameters and of virtual object features to sound parameters is done at similar abstraction levels.

2.1 Abstraction Levels

Our aim is to address the fact that hand movements, virtual object parameters such as shape, and sound are all multidimensionally parameterized at various levels of abstraction. We hope that explicit access by the user to these levels of abstraction will facilitate design and use of a mapping of human movement to sound. We also

hope that a mapping will be faster to learn when movement features are mapped to sound features of the same abstraction level. However, as stated in the introduction, the mouse and keyboard capture only a specific range of hand movements, so that computation of gestural expression at levels of abstraction other than the physical level is always based on that specific range of movements. Hence, the number of gestural expressions represented at other levels of abstractions is only a small part of the full range of gestural expressions. On the contrary, by presenting the input device to the user as a virtual object that can be programmed to respond to almost any hand movements, representations at other levels of abstraction that are of interest become available.

2.2 Mapping Examples

In terms of relating the parameters of a virtual object to sound, a mapping based on the real, physical world may be easy to understand, but will not provide suggestions for the control of abstract, higher level sound parameters. Nevertheless, if the object were taken as a sound radiating object, a simplified mapping of the virtual object's position and orientation to sound parameters could involve mapping left/right position to panning, front/back or depth position to reverb level, angle of the object's main axis with respect to the vertical axis to reverb time and virtual object volume to loudness. Mapping virtual object shape parameters to timbral parameters is less easily based on the physical world, but could involve mapping the length of the main, i.e. longest, axis to flange amplitude, transverse surface area or object width to chorus depth and curvature or "roundness" to brightness. Brightness is used as a parameter in a perceptual sound synthesis model [12]. While we are currently investigating mapping to sound we are interested in applying the developed interaction methodology to other domains, such as the editing of texture, color and lighting of graphical objects.

2.3 Explicit vs. Implicit

The above approach involves explicit knowledge of the mapped parameters, however, another approach would involve implicit knowledge. Such an approach can be implemented with neural networks, which require that the user "teach" the system which hand movements it should map to the sound parameters of interest [3]. In both ways the system can be adapted to the user's preferences, but the implicit method does not provide any guidelines when designing a mapping, which may be an advantage in situations where the user already has an idea of how a sound should change when making a movement but a disadvantage in other situations where guidelines for the design of a mapping are desired or when presets needed.

3 Implementation

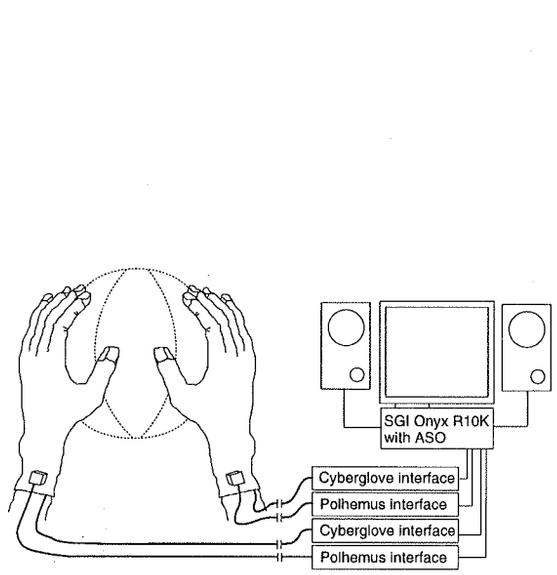
In the development of our prototyping environment we aimed to facilitate quick prototyping and experimentation with a multitude of gestural analysis methods by creating a set of tools that facilitate the computation of various gesture features and parameters.

3.1 Hardware Environment

Figure 2 a) illustrates the hardware device setup. We use an SGI Onyx with 4 R10000 processors and audio/serial option (6 serial ports) to interface two Virtual Technologies Cybergloves (instrumented gloves that measure hand shape - see also [9]) and a Polhemus Fastrak (a sensor for measuring position and orientation of a physical object, such as the human hand, relative to a fixed point). A footswitch enabled "holding" of the virtual object. While the Cyberglove is probably one of the most accurate means to register human hand movements, we have found accurate measurement of thumb movement difficult due to the fact that the sensors intended for the thumb do not map to single joints but to many joints at the same time. Nevertheless, with a sufficiently sophisticated hand model it is possible to reach acceptable accuracy for our purposes, but the calibration is tedious as well as individual specific. For graphic display of the hands and the virtual object we use OpenInventor software.

3.2 Software Environment

We use a visual, interactive programming environment for real-time sound synthesis and algorithmic music composition called Max/FTS [10] (figure 2 b)). We chose Max/FTS as our platform due to its real-time computation and visual programming capabilities, the access to various synthesis models implemented as editable patches and the fact that it runs on an SGI, thus needing no special sound cards. Max/FTS functionality is extended by linking in dynamic shared objects (DSO) at run-time. In order to facilitate quick and easy prototyping of various gestural analysis computations and allowing for application of the computations to different bodyparts we have developed new Max/FTS objects. We exploit the strong datatype checking of Max/FTS to introduce new datatypes such as *position* and *orientation* for geometric and kinematic computations as well as a *voxel* and *hand* datatype. This way the user cannot apply objects at an inappropriate abstraction level, which would be possible if computed values were only represented as number data types. Real-time performance was achieved by using memory that was shared between two Max/FTS applications, one for sound synthesis and the other for remaining computations, to



(a) Hardware devices used. The dotted lines represent a virtual surface which the sound designer or sound composer manipulates.

(b) Typical example of geometric computing using the new Max/FTS *position* and *orientation* objects.

Figure 2: The development environment.

exchange data. The available sound parameters in our simplified synthesis model were loudness, spatialization parameters panning, reverberation time and level and timbre specific parameters attack time, release time, flange index, chorus depth, frequency modulation index, low/mid/high-pass filter amplitude and frequency.

3.3 New Max/FTS objects

The new Max/FTS software object we have programmed are listed below with reference to figure 1.

- **Unix character device drivers** - Objects for interfacing peripheral devices that communicate using one of the SGI's serial ports (*serial*), and for communication between Max and other processes using TCP sockets (*client*) or shared

memory with semaphores (*sms* and *smr*). We are using the *sms* object amongst others to communicate data to an OpenInventor 3D graphics server to display the acquired hand shape, hand position and hand orientation data as a graphic hand as well as to display the virtual object graphically.

- **Sensor interfaces** - These objects are peripheral (character) device interfaces that implement a specific protocol, such as for the Cyberglove and Polhemus sensors (*cyberglove* and *polhemus*).

- **Geometric computation** - New datatypes *position* (x, y and z position as floats in one data structure) and *orientation* (a data structure containing a rotation matrix, euler angles and angle-axis representations of orientation) were defined for this group of objects to facilitate computations such as distance between points ($+P$, $-P$), scaling ($*P$), dot product ($dotP$), cross product ($crossP$), norm ($normP$), magnitude ($\|P$), rotation ($O * P$ and $*O$), frame of reference switching (TO) etc. and their derivatives. Figure 2 b) shows a Max patcher that takes a list of *positions*, projects these on the plane that best fits them, then rotates the plane (with projected *positions*) and eliminates z. This patcher enables subsequent fitting of 2D parametric curves to the x-y coordinates (e.g. the x-y coordinates computed from the *positions* marking the thumb and index fingers).).

- **Geometric structure computation** - At this level the computations do not involve just points with a position and an orientation but involve volume elements (represented as a new datatype *voxel*). Voxels may be ordered and linked in a specific manner so as to represent for instance human anatomical structures or otherwise shaped objects. A new datatype *hand* is used to represent a human hand. The object *geoHand* computes an ordered list of *voxels*, packed as a *hand* that are relative to the Polhemus transmitter from Cyberglove joint data and Polhemus receiver position and orientation data. This computation could also be done using the geometric computation objects in a patcher, but it will be computationally more expensive.

- **Hand feature computation** - Objects for the computation of hand features such as the distance between thumb and index fingertips and the distance between left and right hand palm are easily calculated using the above geometric computation objects in a patcher. We also made objects for computation of the orientation of a plane (*plane*, see also figure 2 b)) such as the palm, the average position of a selected number of points of the hand ($avgP$) and for computation of features based on the path of a selected point of the hand ($bufP$). We intend to make other objects for the computation of features such as finger and hand curvature using a curve fitting algorithm, estimated grip force using a grasping taxonomy etc..

- **Virtual object processing** - We have programmed *sheet*, a physical model of a sheet of two layers of masses connected through springs (see figure 3 a)). The four corners of the sheet function as the control points and can be stuck, or clamped to e.g. the index and thumb tips of both hands. Similarly, we have programmed *balloon*, a physical model of a single layer of masses positioned in sphere-like form (see figure 3 b)).

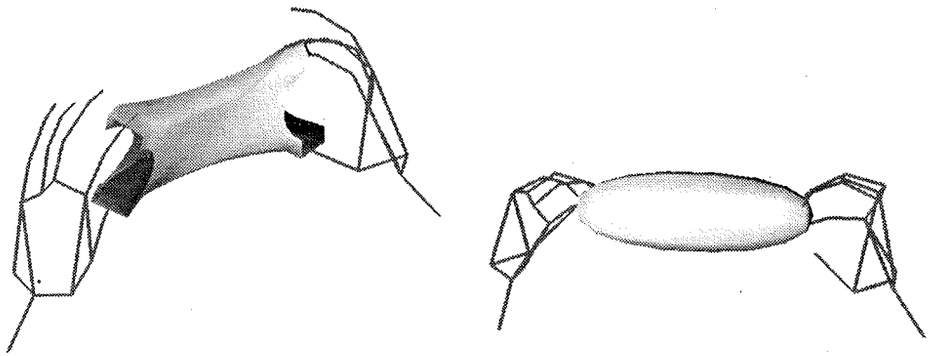
- **Virtual object feature computation** - Many of the geometric computation objects and some of the hand feature objects can be used to compute virtual object features. Other types of virtual object feature computations involve superquadrics, a mathematical method to describe a wide variety of shapes with two parameters specifically related to virtual object (vertical and horizontal "roundness" or "squareness") and 9 others for size, orientation and position of the virtual object. Based on [11] we have programmed a *superquadric* object that fits a virtual object described in terms of superquadric parameters to a set of *positions*. As the fitting process is iterative, it is computationally expensive and as yet too slow (order of 100 ms) for real-time control of timbre. A simpler approach is implemented in a feature computation object *ellipsoid* which fits only a virtual object described in terms of ellipsoid parameters (i.e. three size parameters) to the list of *positions* and computes within real-time. For 2D curve fitting we have programmed a *polynomial* object, which will approximate a list of *positions* by a polynomial of at most second degree. The $kappa2D$ and $kappa3D$ objects compute the curvature of a 2D curve respectively 3D surface. We are further studying 2D and 3D Fourier transforms for use as virtual object features [1].

4 Virtual Object Manipulation

We have created two types of virtual objects, and experimented with manipulation methods based on the pragmatics of claying. The pragmatics of claying consisted of manipulation of the virtual object permanently stuck to (a selected number of) joints and manipulation by "touching" the virtual object. A footswitch enabled selection of either manipulation method.

4.1 Manipulating a Rubber Sheet

We have experimented with the manipulation of a virtual object with a shape and behaviour of a rubber sheet. The object *sheet* was used to compute the virtual object



(a) Sheet clamped to the index and thumb tips of both hands.

(b) Balloon clamped to both hands.

Figure 3: Examples of manipulating virtual objects.

voxels. The *voxels* of the tips of the index and thumb fingers were used as the *voxels* of the four corners of the virtual sheet. Thus, rotating, but not moving, the finger tips would result in bending of the virtual sheet. The computations could be completed near to real-time if a suitably low number of virtual masses was chosen. Fingertips of both hands or one hand could be clamped to the corners of the sheet (figure 3 a)). If only one or none of the hands was clamped, the other hand or both hands could "touch" the sheet.

4.2 Manipulating a Balloon

We have also experimented with the manipulation of a virtual object with a shape and behaviour of a rubber balloon. Manipulation of this virtual object occurred through the movement of any of the *voxels* that make up a hand. When the virtual object was clamped to the hands, a superquadric shape was fitted to the *positions* of these *voxels*. When switching from clamping to "touching", the virtual object surface *positions* were used as the starting positions for a physical simulation of a rubber balloon. Joints of both hands or one hand could be clamped to surface points of the balloon (figure 3 b)). If no hand or only one hand was clamped, the other hand(s) could "touch" the sheet.

5 Evaluation

In our pilot studies to date, pitch and duration of the sound were either fixed or pre-programmed in a MIDI sequence. Mapping virtual object height to pitch could be "intuitive". However, due to measurement latency in the acquisition of the height of the virtual object, controlling pitch in this way was ineffective. Instead the height was mapped to mid-pass filter amplitude. We found that when mapping a virtual object feature to a sound parameter, offsetting and scaling the value of the virtual object feature required some arbitrary heuristics based on workspace dimensions etc.. A solution without such heuristics would be preferred. In the following two subsections we discuss the manipulation and mapping of the sheet and balloon virtual objects.

5.1 The Rubber Sheet

We evaluated the rubber sheet using the mapping of virtual object position and orientation as described in the section on mapping examples and the following mapping of virtual object shape. Average length of the sheet, along the axis between left and right hand was mapped to flange index. Width (i.e. axis between index and thumb) of the sheet was mapped to chorus depth. A measure of the average curvature, computed with *kappa3D*, was mapped to the frequency modulation index. The angle between left and right edge of the sheet was mapped to vibrato. The mapping of virtual object position and orientation took only a few moments to get used to. As for shape, it was found that curvature was difficult to control given the fact that only the four corners of the sheet could be manipulated. Normally, one positions fingers in various ways on the surface to control the curvature. The length was easiest to control, then the width and the angle between left and right edge of the sheet. Manipulation of the sheet itself with the indexes of both hands clamped to the sheet was very natural - there was very little effort required to master control.

We also tried manipulation with the left hand only, similar to the familiar keyboard modulation wheel but with increased functionality. This form of manipulation with the left corners of the sheet clamped to the left hand index and thumb and the right corners of the sheet fixed in space, allowed gestures to be more expressive than typical keyboard modulation wheel gestures and hence provided for a more dramatic performance effect. In addition, the reference point (the right corners) could easily be moved, so as to accomodate rapid changes in the zero-point of any modulations given different musical context, by clamping the right corners to the right hand temporarily. Manipulation using "touching" only was difficult, mainly due to the lack of tactile feedback.

5.2 The Rubber Balloon

We evaluated the rubber balloon using the mapping of virtual object position and orientation as described in the section on mapping examples. As in the case of the sheet, this mapping worked equally well. We evaluated the following mapping of virtual object shape. The length of the main, i.e. longest, axis of the balloon was mapped to flange index and the cross sectional surface area was mapped to chorus depth. The objects *ellipsoid* and *superquadric* were used to compute the virtual object features. Informal evaluation showed this mapping to work well. "Roundness" of the object, estimated using the bending of the fingers was mapped to the frequency modulation index. Length was easiest to control and then cross sectional surface area and "roundness". This aspect of the mapping was more difficult to use due to occasional "manipulation cross-over", i.e. when the cross-sectional surface area was increased the balloon would also become more rectangular instead of ellipsoid, without the sound sculptor intending it. Another problem with this mapping is that it is unclear how the sound sculptor conveys "roundness" of the virtual object when changing the shape of his or her hands, due to the fact that normally not the entire hand is touching an object's surface. Any sphere-like object can be held with anywhere from a few finger tips to the entire hand. In addition, due to the fitting process, jitter occurred, i.e. the virtual object would at times jump from cube-like to ellipsoid without any significant hand motion, resulting in unpredictable sound variations. Possibly this can be circumvented by algorithm and patch adaptations.

In general, most test subjects informally felt more comfortable working with the balloon than with the sheet using our mappings. Manipulation using the left hand only, with the balloon entirely clamped to the hand was not as natural as in the case of the sheet, most likely due to a missing reference point (the right corners in the case of the sheet) in our case (the balloon could also be fixed in space at one end, but we haven't implement this yet) and hence more difficulty in manipulating the shape of the balloon. Nevertheless manipulation of the position and orientation of the virtual object was effective, but the same effect may have been achieved without the presence of the virtual object. For similar reasons as in the case of the sheet, manipulation using "touching" only was difficult.

5.3 Summary of Evaluation

In summary, informal evaluation of manipulating the sheet and balloon virtual objects showed that the balloon appeared to provide more natural interaction than the sheet for the mappings we provided and that the left-hand-only manipulation method could provide a useful alternative to the standard keyboard modulation wheel in sit-

uations where the right hand is needed for pitch control using e.g. a keyboard. The control of virtual object shape often required some effort to master due to the need for unusual movements to achieve common manipulation results or due to unexpectedly ambiguous gestures. Thus, we need to define virtual object features more carefully, taking into account manipulation pragmatics. While mapping position and orientation proved easy to use, the mapping of virtual object shape to a variety of timbral parameters, with no obvious analogy offered to the user, required learning to achieve desired acoustic feedback naturally using the manipulation methods.

6 Conclusions

We have succesfully prototyped sound sculpting, an application of virtual sculpting as a new real-time interaction method for sound design and musical performance. We have implemented sound sculpting allowing for a multiple abstraction level mapping strategy by creating new software objects and data types for Max/FTS to interface a variety of sensors and to compute hand movement features and virtual object features. To date we have evaluated mapping of a virtual rubber sheet and of a virtual rubber balloon to sound spatialization and timbre parameters. It was found that the more physical analogies are used for a mapping the faster the mapping can be learned. This requires more research in order to map virtual object features to timbre naturally. While both hands can be used for manipulation, left-hand-only sound sculpting may be a useful replacement for and extension of the standard keyboard modulation wheel. To reduce learning time we need to address more manipulation pragmatics in mapping hand movement to sound and more carefully define movement features to map to virtual object parameters. We hope to apply the developed interaction methodology to other domains, such as the editing of texture, color and lighting of graphical objects.

Acknowledgement
This work was supported by the Advanced Telecommunications Research Institute in Kyoto, Japan.

References

[1] Ch. Brechbuhler, G. Gerig, and O. Kuhler, "Parametrization of closed surfaces for 3D shape description," *Computer Vision and Image Understanding*, Vol. 61, No. 2, March, pp. 154-170, 1995.

[2] Paul Kabbash, William Buxton and Abigail Sellen, "Two-Handed Input in a Compound Task," *Proceedings of CHI '94*, pp 417-423, 1994.

[3] S. Sidney Fels and Geoffrey E. Hinton, "Glove-TalkII: Glove-TalkII: A neural network interface which maps gestures to parallel formant speech synthesizer controls," *IEEE Transactions on Neural Networks*, Vol. 9, No. 1, pp. 205-212, 1998.

[4] S. Sidney Fels and Geoffrey E. Hinton, "Glove-Talk: a neural network interface between a data-glove and a speech synthesizer," *IEEE Transactions on Neural Networks*, Vol. 4, No. 1, pp. 2-8, 1993.

[5] Tinsley A. Galyean, "Sculpting: An Interactive Volumetric Modeling Technique," *ACM Computer Graphics*, Vol. 25, No. 4, (SIGGRAPH '91, Las Vegas, 28 July - 2 August 1991), pp. 267-274, 1991.

[6] Axel G. E. Mulder, "Getting a GRIP on alternate controllers: Addressing the variability of gestural expression in musical instrument design," *Leonardo Music Journal*, Vol. 6, pp. 33-40, 1996.

[7] Axel G. E. Mulder, "Hand gestures for HCI," *Technical Report, NSERC Hand Centered Studies of Human Movement project*, Burnaby, BC, Canada: Simon Fraser University, 1996. Available through the WWW at http://www.cs.sfu.ca/~amulder/personal/vmi/HCI-gestures.htm

[8] Axel G. E. Mulder, "Virtual Musical Instruments: Accessing the Sound Synthesis Universe as a Performer," *Proceedings of the First Brazilian Symposium on Computer Music*, (Caxambu, Minas Gerais, Brazil, 2-4 August 1994, during the 14th Annual Congres of the Brazilian Computer Society), pp. 243-250, Belo Horizonte, MG, Brazil: Universidade Federal de Minas Gerais, 1998. Available through the WWW at http://www.cs.sfu.ca/~amulder/personal/vmi/BSCM1.ps.Z

[9] Axel G. E. Mulder, "Human Movementq Tracking Technology," *Technical Report, NSERC Hand Centered Studies of Human Movement project*, Burnaby, BC, Canada: Simon Fraser University, 1994. Available through the WWW at http://www.cs.sfu.ca/~amulder/personal/vmi/HMTT.pub.html

[10] Miller Puckette, "FTS: A real time monitor for multiprocessor music synthesis," *Computer music journal*, Vol. 15, No. 3, pp. 58-67, 1991.

[11] Franc Solina and Ruzena Bajcsy, "Recovery of parametric models from range images: the case for superquadrics with global deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 2, February, pp. 131-147, 1990.

[12] David Wessel, "Timbre space as a musical control structure," *Computer music journal*, Vol. 3, No. 2, pp. 45-52, 1979.

Objects in a Virtual Space: A Comparative Analysis beetween Image and Sound Spatial Representation and Synthesis

S. NATKIN
CEDRIC, CONSERVATOIRE NATIONAL DES ARTS ET METIERS
292 rue St Martin 75141 Paris Cedex 03, France
natkin@cnam.fr

1. Goal of the research

This paper is a comparative analysis of the spatialisation process in virtual space simulation both from the visual and audio point of view. It is the starting point of a research project of the CEDRIC laboratory multimedia team. The main purpose of this resarch is to find design principles for multimedia synthesis tools, that is to say tools able to create and manage multimedia virtual objects. The users of such tools should be able to define relationship between sound and visual characteristics and to play with sound and visual perception phenomena.

The spatialisation process is the last step in the synthesis of a 3D scene (Figure 1). It is mainly the computation of the object interactions, considered as light and sound sources or reflectors, in a virtual space. Comparing audio and image spatialisation principles has several practical goals:

- It is a step in the analysis of adequacy of each classes of algorithm to different classes of applications (from movies post processing to virtual reality).
- It allows to specify software and hardware components efficient in both fields.
- It provides a reference for the definition of audio and image objects descriptors for virtual scene specification languages (such as VRML (vrml) or JAVA3D (Java))
- More generally it is a first step into the creation of a multimedia synthesis tool.

As an example, Figure 2 is an informal representation of a virtual scene defined in a object programming environment (Jacobson 1993). This scene contains two objects: a wall and a man. Each object is defined as a new instance of a generic class, which includes some audio and image specifications of the object (source or reflector for example). The object "man" has two "animation" methods one four the sound and one for the image. This allows, for example, an animation such that the man's image leaves the scene in one direction and the sound leaves the scene on another one.

The image spatialisation process is well defined and a classical survey can be found in (Foley 1997). There have also been numerous papers on sound spatialisation and its relation to psycho acoustic (see for example (Blauert 1996) (Jot 1997), (Jot 1998) or (Julien 1994)). To our knowledge a very little work have been done in the comparison and the synthesis of these two fields. This paper is a first step in this direction. It is organized as follows: In the next section we give a more precise definition of the spatialisation process, the third section is devoted to the comparison of the physics and perceptual similarities and differences used in the design of sound and image synthesis algorithm. The last section presents a classification of algorithms in both domains. The presentation of this paper will be illustrated with several sound and video examples.

2. The spatialisation process

Numerous parameters related to sound and light propagation define objects in a virtual scene. An object is first defined by its current shape and position, then it can be considered either as a sound (resp. light) source or reflector. The absorption and reflexion characteristics of a reflector must be described either by a formal model or using a recorded representation (we call such a representation a texture in the sequel). The specification of a source object includes the directionnality and the signal. For example a light source can be defined just by a three component (RVB) constant light spectrum and the diffusion solid angle. The walking man of figure 1 can be considered as an omnidirectionnal sound source, which signal is determined by a recorded sound object (i.e. a sound file walking.aiff). To complete the scene description one must add a model of the medium propagation model (wave celerity and absorption as a function of the spectrum). This medium specification is generally implicit for light (except for foggy weather scene) and explicit for sound.

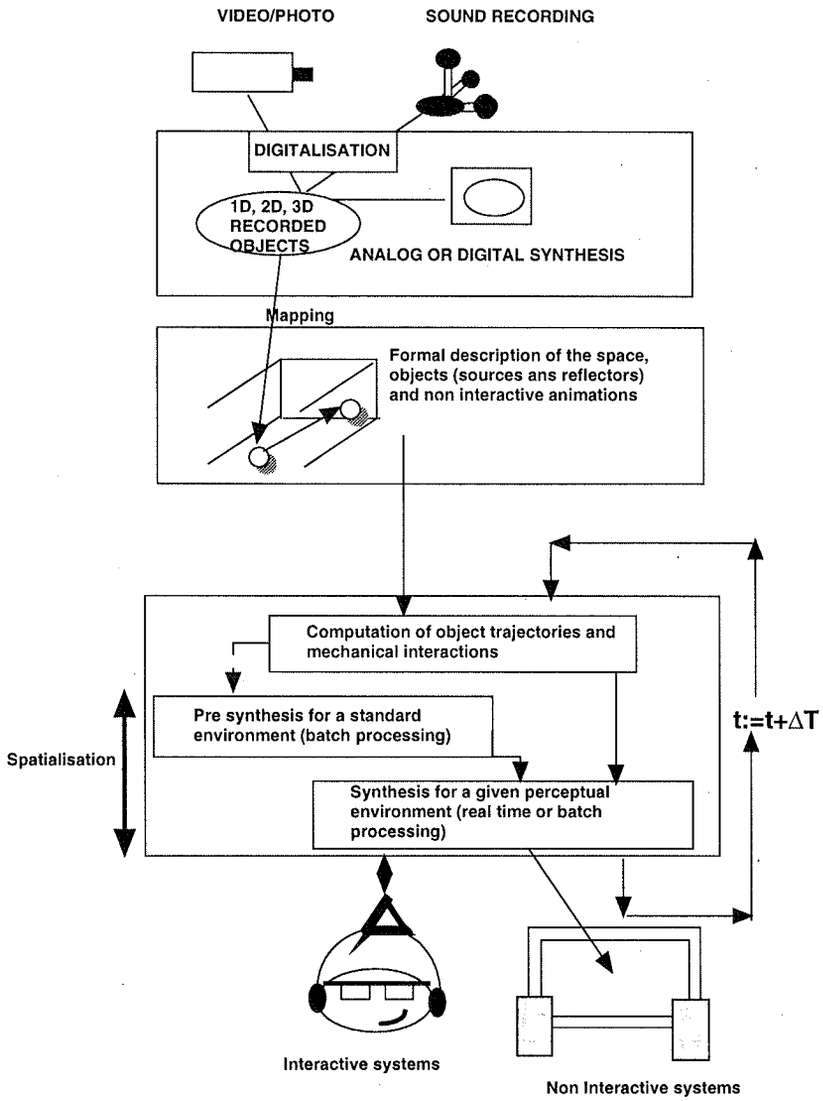


Figure 1: Sound and image synthesis process

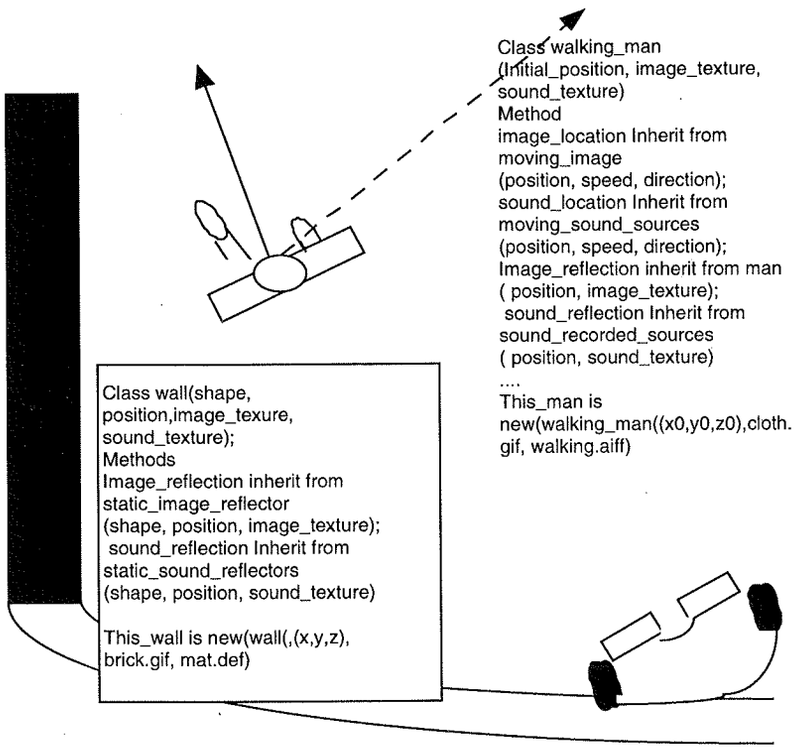


Figure 2: Specification of a virtual scene

The South East corner of figure 2 represents a spectator of the virtual scene which see and hear the virtual scene through a perception display. It can be a headset with goggles or loudspeakers and a screen. The spectator may be either passive or interacting with the virtual scene (at least by moving his head).

The spatialisation process computes the sound and the images as they are diffused through the perception display, considering the time evolution of the scene and the interactions between objects in the light and sound propagation. Figure 3 illustrate this computation, which is essentially composed of a "ligthning" algorithm (Foley 1997) and a projection algorithm according to the current location of the spectator. The last step (post processing) is the adaptation of the results computed to the perception display.

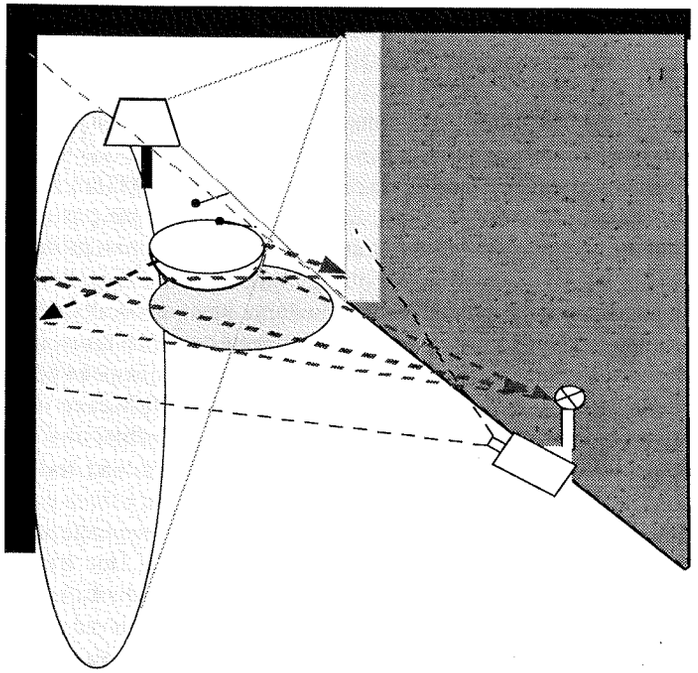


Figure 3: Illustration of the spatialisation process

3. Comparisons between sound and images spatialisation principles

Spatialisation relies on a subtle merging between physical and psycho perception principles to define computability efficient algorithms. At the most formal level, wave physics, the two phenomena can be considered as quite similar as they are described by the same kind of partial differential equations. This is the main reason which allows to use similar classes of model. For example optic and acoustic geometry use the same "Descarte's laws" of reflection which leads to the well known ray tracing class of algorithm (Whitted 1980),(Foley 1997),(Vorlander 1989).

But formal or numerical solution of an accurate physical model can not be obtained in a reasonnable amount of computation, even for rather simple dynamic scenes. Moreover, the goal of the spatialisation process as defined in the last section is not to simulate accurately the real world (as for example in virtual acoustic) but to built a "good" perception of the virtual scene according to the design goals and the diffusion display used.

Hence the spatialisation process uses characteristics of the perception of light and sound, which are quite different (sampling rate, directionnality, accuracy...) related to the wave propagation characteristics.

Many factors can be considered comparing the spatialisation algorithms. It concerns mainly:

- The difference between the celerity of waves (330 m/s vs 300 000 km/s).

  Hence the spatialisation of an image does not depend on the past (light propagation is instantaneous) and the spatialisation of the sound is a time convolution process.
- The perception of waves attenuation and diffraction

  Sound attenuation in a homogeneous medium must generally be considered. Light attenuation in a homogeneous «transparent» medium need not to be taken into account (except if needed or for special effects). Diffraction of sound objects is essentially considered through the variation of the celerity as a function of the spectrum. Diffraction in light propagation through transparent objects determines the computation of the ray direction.
- The wavelenghts related to the size of source and reflectors, and the accuracy of human discrimination (mm vs nanom).

The main aspects of the object characteristics used in the image spatialisation process depend rather simply on the wave spectrum. The RVB model, for example, relies on the filtering characteristics of human vision. Accurate sound spatialisation relies on the whole spectrum. We hear sound sources, reflectors are mainly perceived as contribution to a reverberation process. In counterpart we see essentialy light reflectors. So sound sources must be modelled precisely, sound reflectors can be defined from a more "global" point of view. Accurate lightening and shadowing of a 3D scene needs a precise definition of reflectors and sources may be modelled in a more approximate way.

Last but not least, one must consider the human anatomy which imply that we see at a given time a half space and hear the whole space.

4. Spatialisation algorithms

Spatialisation algorithms can be classified in term of complexity which increases with the precision of the physical model used as the algorithm basis. The same classification can be used in the audio and image processing fields. In this section we present a classification and discuss briefly each class of algorithm in both fields.

The simplest spatialisation process is just a mean value computation of the energy diffused in the scene. The mean value parameter is more related to a perceptual than a physical parameter. The ambiant lightning algorithm for example does not consider light sources but a homogeneous field in which each reflector has a given reflexion coefficient. More sophisticated models (Phong and Warn lightening algorithms (Bui-Tuong 1975), (Warn 1983)) consider the

superposition of the ambiant light field and the direct effect of light sources. The complexity of this class of algorithm is proportional to the product of the number of light sources by the number of visible pixels. So the lightning algorithm follows a hidden surfaces computation which complexity is roughly proportionnal to the number of reflectors and which depends on the scene geometry and the spectator position. The computation must be repeated independantly for each image (25 to 30/s). As the light sources and the spectator position move slowly and the the light spectrum of each source is generally constant in the time, the computation can be optimized by an interpolation of an image from the previous one.

The corresponding sound spatialisation algorithm consists of the superposition of the direct sound and a statistical response of the room the later beeing synthesized by the use of basic reverberation algorithms (see [Jot 1997 2) for a survey and a presentation of new results). For example the IRCAM Spat processor combines the direct sound, the early reflexions and late reverberation. In this tool, the room is not necessarely defined by its geometry but by simple perceptual factors. The complexity is proportionnal to the number of sources and listeners, but is essentially dependant of the "memory" of the process which is inherent to the fact that the sound emitted by each source is a time function.

It is interesting to note that this class of algorithms were found by graphic computer scientists for the image and initially based on perceptual factors. In counterpart the sound algorithms were defined by acousticians and directly correlated to the physical analysis of the phenomenon.

The second class of algorithm relies on geometrical optic and acoustic. The well known ray tracing (Whitted 1980) (and image source in the sound field) model is the main basis principle of this class. Ray tracing is used in many other fields of simulation such as physic of particles or billiards. The algorithm determines the trajectories of sound and light rays from each source to each spectator. Rays are reflected on each reflector according to the Descarte's laws. At each reflection point the part of the signal which is absorbed and reflected must be computed. Hence in both fields the algorithm relies on a geometrical description of objects. This description can be rather simple in sound computation and must be very precise in the lightning algorithm. Light rays are generally geometrical lines, sound rays can have a "volume". The main difference between the two fields is related to the directionnality of the perception. This allows in image synthesis (Figure 4), to consider only the rays which start from the spectator's eyes and reach each monitor pixel, which is obviously impossible in audio spatialisation. Of course the difference on the memory of the two computations is the same than in the mean value class. This leads in the audio field to the source image algorithm (Vorlander 1989). Reflected rays are replaced by virtual sources situated behind the walls of the room. The contribution of each virtual source is delayed with respect to the original one (direct sound) by an amount of time which model the sound propagation. This simplification is efficient when the sources and the spectator have a static position. If the spectator moves, the contribution of each virtual source

has to be periodically computed, if the source mooves both the position and the contribution of each virtual source must be computed for each source location.

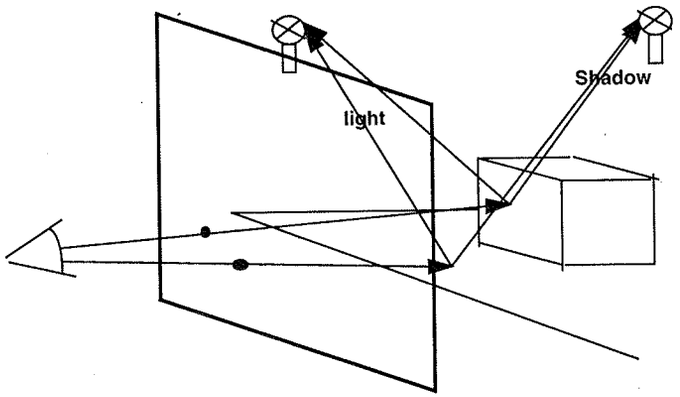


Figure 4: Recursive Image Ray tracing

This geometric class of algorithm is probably the one where the comparative analysis can lead to the most interesting results in term of common description of multimedia objects and software components. But, up to now, these algorithms are still too complex for a real time computation.

The last class of algorithm relies implicitely or explicitely to the numerical solution of the Dirichlet problem. We include in this class Radiosity algorithms, which replace the partial difference equations by a linear algebraic system which express the energy conservation and solves implicitely the Laplacian field. The algorithm are quite similar in audio and image synthesis and are used for "static" computation (still image in computer graphics, impulse response in virtual acoustic). So it is not yet usefull for the spatialisation process as defined in this paper.

5. Conclusion

This paper is a first comparative analysis of the spatialisation in audio an image synthesis. Numerous points have been omitted, such as the generation according to the reproduction display or a state of the art on virtual scene specification languages. As a position paper, we hope to have show the interest to this field of research, which is, to our point of view, the basis for the design "real" multimedia tools.

Acknowledgements

This paper is the results of many discussions with Jean Marc Jot from IRCAM. I would like also to thanks Cecile Le Prado, composer, users of sound spatialisation tools and my wife, for her critical observations on the first draft of this paper, Pierre Cubaud and Jean Marc Farinone of CNAM for there usefull remarks and corrections.

6. References

Blauert 1996 Blauert J, Spatial Hearing: The Psychophysics of Human Sound Localisation, MIT Press, 1996

Bui-Tuong 1975 Bui-Tuong, Phong, *Illumination for Computer Generated Pictures*, CACM, 18 V6, June 1975

Cohen 1995 M. Cohen, E. Wensel, *The Design of Multidimensionnal Sound Interfaces*, Technical report 95-1-004, University of Aizu, February 1995

Foley 1997 J.D. Foley et als, *Fundamentals of Computer Graphics*, 4th Ed. Addison Westley, Reading 1997

Jacobson 1993 Ivar Jacobson, *Object Oriented Software Engineering*, Second Edition, Addison Westley 1993.

java http://www.javasoft.com/products/javamedia/3D/

Julien 1994 P. Julien, O. Warusfel, *Technologies et perception auditive de l'espace*, Cahiers de l'Ircam 1994, http://varese.ircam.fr/articles/textes/ Julien94/

Jot 1997 J.M. Jot, O. Warusfel, *Techniques, algorithmes et modèles de représentation pour la spatialisation des sons appliquée aux services multimedia*, IRCAM research report 1996, http://varese.ircam.fr/articles /textesJot97a/

Jot 1997-2 J.M. Jot, L. Cerveau, O. Warusfel, *Analysis and Synthesis of Room Reverberation Based on a Statistical Time-Frequency Model*, AES 103rd Convention, Sept 1997, New York, USA

Jot 1998 J.M. Jot, *Real Time spatial processing of sound, music multimedia and interactive human-computer interface*, to be published in ACM Multimedia System Journal, special issue on audio and multimedia.

Vorlander 1989 M. Vorlander, *Simulation of the transient ans steady state sound propagation in rooms using a new combined ray tracing/image source algorithm.*, J. Acoust Soc Am. 86 (1) July 1989

vrml http://www.vrml.

Warn 1983 D.R. Warn, *Lithening Control for Synthetic Images*, Proc of SIGRAPH 83, In Computer Graphics 18 V3 July 1984.

Whitted 1980 Whitted T., *An improved illumination method for shared display*, CACM 23 (6), 1980

Traité des objets musicaux revisited

CARLOS PALOMBINI
Visiting Lecturer
Departamento de Música
Universidade Federal de Pernambuco
CAC
Av. Acadêmico Hélio Ramos s/n
Cidade Universitária
Recife PE 50740–530
Brasil
Palombini@altavista.net

Abstract

While Pierre Schaeffer's taste for paradox puzzles readers, one of the greatest paradoxes of *Traité des objets musicaux* is that it remains one of the most cited and least read books of the electroacoustic literature. Clearly, no one has thought out the aesthetic implications of the analogue sound medium with Schaeffer's insight and depth. And yet computer music, because it has not always paid enough attention to this reflection, sometimes finds itself conceiving of the digital domain with the conceptual apparatus of handwork technology. The synopsis and contextualization that follow pay tribute to the fiftieth anniversary of *musique concrète*.

The original version of this paper was written for *The Twentieth Century Music Avant-garde: A Biocritical Sourcebook* (Greenwood Press, Westport CT, USA, forthcoming Dec. 1999).

Between 1945 and 1953, writes Mitcham (1994), technology took the world stage in defiance of the human mind that fostered it up: A-bomb (1945), ENIAC (1946), kidney transplant (1950), H-bomb (1951), UNIVAC (1951), DNA (1953). Between 1954 and 1962

the new powers were put to use within traditional frameworks, with increasingly conflictive results: USS *Nautilus* (1954), birth control pill (1955), MASER (1955), *Sputnik I* (1957), integrated circuit (1959), LASER (1960), *Vostok* (1961), *Mariner 2* (1962). The period comprised between 1963 and 1972 was one of the most creative in the history of science and technology policies. Human, political, and economic frameworks were adapted or altered. Initiatives in assessment and control were taken: limited nuclear test ban treaty (1964), Harvard University Programme on Technology and Society (1964), *Humanae vitae* (1968), Greenpeace (1969), Earth Day (1970), Stockholm Conference (1972), *The Limits to Growth* (1972).

In 1951 the French Radio presented the Groupe de Recherches de Musique Concrète, which at the time consisted of Pierre Schaeffer,[1] the engineer Jacques Poullin, and the composer-percussionist Pierre Henry, with the first purpose-built electroacoustic music studio ever. The collaboration between Schaeffer and Poullin, in its fourth year, was resulting in a three-track tape recorder, a machine with ten playback heads to replay tape loops in echo (the Morphophone), a keyboard controlled machine to replay tape loops at twenty-four pre-set speeds (the keyboard, chromatic, or Tolana Phonogène), a slide-controlled machine to replay tape loops at a continuously variable range of speeds (the slide, continuous, or Sareg Phonogène), and a device to distribute live an encoded track across four loudspeakers, including one hanging from the centre of the ceiling (the potentiomètre d'espace). Output from 1951 to 1953 comprised *Étude I* (1951) and *Étude II* (1951) by Boulez, *Timbres-durées* (1952) by Messiaen, *Étude aux mille collants* (1952) by Stockhausen, *Le microphone bien tempéré* (1952) and *La voile d'Orphée* (1953) by Henry, *Étude I* (1953) by Philippot, *Étude* (1953) by Barraqué, the mixed pieces *Toute la lyre* (1951) and *Orphée 53* (1953) by Schaeffer/Henry, and the film music *Masquerage* (1952) by Schaeffer and *Astrologie* (1953) by Henry. In 1954 Varèse and Honegger visited to work on the tape parts of *Déserts* and *La rivière endormie*.

In 1953 the Groupe de Recherches de Musique Concrète of the Radiodiffusion-Télévision Française rallied elektronische Musik, music for tape, and 'exotic music' under the banner of experimental music to compare methods and establish complementary research programmes (see Palombini 1993). Written in 1953 and published in 1957, 'Vers une musique expérimentale' minimized frictions. Considering that tonal relations were inherent to the construction and technique of Western instruments, Schaeffer in principle objected to serial methods as applied to traditional instruments, but he observed that, in practice, the *listening* to such pieces could be validated by a certain technique of hearing. Considering that when applied to sound qualities other than pitch the series would lose its negative character and open to new sounds the domains of tradition, Schaeffer in principle

[1] 'Engineer by necessity, writer by vocation, composer by chance'... 'Neither researcher, nor composer, nor writer'... 'Writer by inclination, musician by heredity, polytechnician by constraint, innovator by complexion'... 'Author of literary texts marked with the concern of style'...

accepted the application of serial methods to complex sounds, but he observed that, in practice, such sounds had little to gain from systematic recourse to serial techniques. The methodological syncretion failed to materialize. In Paris and in Cologne, manipulated samples and electronically-produced sounds amalgamated into Henry's *Haut voltage* and Stockhausen's *Gesang der Jünglinge* in 1956. In 1957 Schaeffer outlined the method of research after musique concrète. *Étude aux allures* (1958), *Étude aux sons animés* (1958), and *Étude aux objets* (1959) illustrated that method. In 1958 the Groupe de Recherches de Musique Concrète was transformed into Groupe de Recherches Musicales. In 1959 Tolana advertised the Phonogène Universel, which dissociated downward and upward shifts in tessitura (spectral transposition) from distension and contraction of dynamic shapes (temporal transposition).

In 1954 Heidegger stated that humans were delivered over to technology in the worst possible way when they regarded it as something neutral; for this conception of it, to which they particularly liked to do homage, made them utterly blind to the essence of technology.[2] Because the essence of technology was nothing technological, essential reflection upon technology and decisive confrontation with it ought to happen in a realm that were, on the one hand, akin to the essence of technology and, on the other, fundamentally different from it. Such a realm was art. But only if the arts were not conceived as deriving from the artistic, if art works were not enjoyed aesthetically, if art were not a sector of cultural activity. Art demanded to be reconducted to the golden age of Greek techne. In 1958 Simondon saw culture as unbalanced because it enshrined the aesthetic object in the world of significations while driving the technical object back into the structureless world of what had no signification but a use. Simondon sought to integrate the machine into the family of human things as a component of a global rebirth of culture. The gap which separated the occidental man from the work of his hands demanded to be bridged. And the activities of the craftsman, simultaneously ancient and modern, provided a model of understanding, employment, and humanization of the machine (Hart 1989).

Pierre Schaeffer delivered *Traité des objets musicaux, essai interdisciplines* in 1966 after fifteen years of labour. The work was dedicated to the memory of his father, whose precept — 'practise your instrument' — the author passed on. *Traité* follows a zigzag course in seven jumps named books. 'Book One' links the genesis of music to the birth of the musical instrument, defined as the causal permanence that engenders organizations of sound *character* (and hence timbre), out of which variations of musical *values* (paradigmatically pitch) appear. 'Book Two' postulates four functions of listening. *Ouïr* (to hear) is to posit iconic (i.e. similarity based) relations between representamen and object (or signifier and signified): on the verge of semiosis, creaks lay dormant in the background

[2] Machover (1984) introduces *Musical Thought at IRCAM* ascribing the diversity of musical outlook there 'also to the neutrality of technology, which offers powerful tools for exploration and creation but does not orient the composer in any particular musical direction.'

noise. *Écouter* (to listen) is to posit indexical (i.e. causal) relations between representamen and object: creaks stand for ungreased hinges. *Comprendre* (to comprehend) is to establish symbolic (i.e. consensual) relations between representamen and object: creaks stand for tempered pitches agreeable to a metrics of successive divisional operations. And because hearing, listening, understanding, and comprehending are lexicalized acceptations of *entendre* — by semantic derivation from the etymological sense, 'to turn one's attention' — the French language allows Schaeffer to construe *entendre* as to hear, listen, understand, and comprehend in mindfulness of one's intention.[3] Thus sounds open themselves up to iconism, indexicality, and symbolism with intent. Reduced listening follows thence as a bracketing of symbolic and indexical relationships such as references to the traditional solfège and to source or causality might afford, whereby the *sonic object* unveils itself as an aggregate of shape and matter qualities. As *ouïr* ebbs *entendre* flows, as *entendre* ebbs *ouïr* flows, and as such movements alternate, sonic things disclose themselves as sonic objects whose intrinsic qualities bespeak details of the sound-producing event and novel abstractive possibilities.[4] 'Book Three' shows the distinct natures of, on the one side, the physical measurements of frequency, time, amplitude, and spectrum, and, on the other, the subjective perceptions of pitch, duration, intensity, and timbre, thus highlighting the perceptual frailty of the soundest parametric construction. 'Book Four' appropriates Husserl, Gestalt, Jakobson, Lévi-Strauss, Merleau-Ponty, Peirce, and Saussure in the interest of musical research. 'Book Five' sets forward criteria to single out sound units from sound continua (identificatory typology) and to select sonic objects where musicalness dwells in posse (classificatory typology). 'Book Six' expounds the method of musical research and outlines seven *criteria* of the morphology of the potentially musical object that are likely to emerge as musical *values* in the context of structurations: mass, dynamics, harmonic timbre, melodic profile, mass profile, grain, and pace. Enlarged for the 1977 reprint, 'Book Seven' comes to the conclusion that no *universal polymorphous musicalness* has arisen from the systematic analysis of sonic objects.

The Solfège of the Sonic Object purports to take, from the practice of sound-producing bodies, to a universal musicalness through a technique of hearing. It comprises a preliminary stage, four operations, and an epilogue. In the preliminary stage, heterogeneous sound-producing bodies are put into vibration by various processes and the resulting sounds are recorded. In the first operation — Typology — sonic objects are singled out from sound continua and selected or discarded according to a musicianly penchant.[5] In the second

[3] Schaeffer exhumes the oppositions *ouïr/écouter,* where *ouïr* signifies the physiological phenomenon, *écouter* the psychological act; *entendre/écouter,* where *entendre* signifies the physiological phenomenon, *écouter* the psychological act; and *ouïr/entendre,* where *ouïr* signifies the physiological phenomenon, *entendre* the psychological act (cf. Littré 1877 and Barthes 1977).

[4] Schafer (1977) notes that, unlike Schaeffer's sonic object, the *soundscape* cannot dispense with causality and meaning.

[5] Typology establishes that the level of complexity of a sonic object is contingent upon the listener's

operation — Morphology — the objects selected are compared, perceptual criteria that make them up are named, and the objects are classed as tokens of such criteria.[6] In the third operation — Characterology — interactions of criteria within a given object are identified and referred to a sound-producing event.[7] In the fourth operation — Analysis — objects evincing a particular criterion are set against the perceptual fields of pitch, duration, and intensity so as to establish cardinal (absolute) or ordinal (relative) scales of criteria. In the epilogue — or Synthesis — new musics are expected to arise, based on reference structures that should play, for each of the seven morphological criteria, a role similar to that played by interval relations and the games of tonality and modality.[8]

The nexus of Schaeffer's research becomes transparent when some avatars of the question concerning the instrument speak their names: 'relay-arts' (1941, 1946), or analogue techniques of sound and image reproduction as instruments of new art-forms; 'noise piano' (1950), or organizing heterogeneous sound-producing bodies into new musical instruments; 'turntable piano' (1950), or analogue techniques of sound reproduction as applied to the conception of a most generic musical instrument; 'cut bell' and 'looping groove' (1950), or analogue sound manipulations as instrumental in the disclosure of the sonic reality; 'pseudo-instrument' (1952), or organizing sonic objects into virtual musical instruments; 'piano law' (1960), or the inverse relation between spectral richness of resonance and incisiveness of attack across the piano tessitura (i.e. the lower the tone, the richer the spectrum and the less incisive the attack, the higher the tone, the poorer the spectrum and the more incisive the attack); 'characterology' (1966), or the systematic investigation of such laws as a means to retrieve the sound-producing event in sonic shapes; 'translation into sound' and 'translation from sound' (1966), or the traditional composer's and the sound recordist's divergent technologies of listening; 'acousmatic listening' (1966), or sound recording as an instrument of poiesis.

In 1936 Benjamin expounded the decline of that unique appearance of a remote reality, however near (the 'aura' of artworks and nature), as a result of the proliferation of mechanical reproduction. In 1954 Heidegger evoked the flow of distanceless uniformity

dissective or integrative intention. Schafer (1977) picks out Schaeffer's term for the smallest autonomous component of a soundscape (but see note 4 above). Cadoz (1984) 'broadens' the concept, applying the term to complex sounds: 'in Schaeffer's book, the notion of an object is associated with elementary sounds.'

[6] Smalley (1986) slices typo-morphology into pieces, splices them into 'spectro-morphology' — 'a preferable term' — and pronounces spectro-morphological thinking 'the rightful heir of Western musical tradition'.

[7] Risset's 1966 discovery — by digital analysis and synthesis — that the brassy character of trumpet tones ensues from a linkage between amplitude increase and upper partials boost fits into the Characterology project.

[8] Lerdhal (1987) purports to lay the foundations of 'an authentic syntax of timbre', likewise modelled on tonality.

where all things were carried away and mixed up; by plane, the radio, the film, television, and the atomic bomb. The bomb and its explosion were the mere final emission of what had long since taken place: the estrangement of Western thought from the thingness of the thing. In the same year, Heidegger depicted the sinking of the object into the objectlessness of the standing reserve under the rule of Ge-stell, the essence of modern technology, according to which everything, including man himself, had become material for a process of self-assertive imposition of human will on things, regardless of their own essential natures (Hofstadter 1971).

Schaeffer's relay-arts instrument pertains to the history of mechanical reproduction, and there is a close resemblance between the two manifestations of *technische Reproduzierbarkeit* as expounded by Benjamin— 'artwork reproduction and the art of film' — and the double role of the relay-arts instrument as expounded by Schaeffer: 'to retransmit in a certain manner what we used to see or hear directly and to express in a certain manner what we used not to see or hear' (see Palombini 1997). In the history of mechanical reproduction, the relay-arts instrument materializes the shift from 'older handwork technology' to that technology which, in the words of Heidegger (1954), 'unlocks, transforms, stores up, distributes, and switches about' the energies of nature, and whose essence Heidegger terms Ge-stell. The 'decline of the aura' — a feat of mechanical reproduction —is intersected by 'the sinking of the object into the objectlessness of the standing reserve'— a feat of Ge-stell — but while the former paves the way for art as political praxis, the latter elicits from Heidegger an invitation to a return to the golden age of presocratic techne. Is this not praxis?

Pythagoras professed cyclic recurrence of events, metempsychosis, and kinship of humans with all living things. Yet he 'carried out scientific investigation more than anyone else' (Heraclitus) and 'was not the lesser of Greek sages' (Herodotus). Upon Pythagoras' death, his followers split into Acousmatics (practitioners of the mystic doctrine) and Mathematics (remarkable scientists). For Schaeffer, music had not sprung from the numeric proportions of intervals. Larousse presented the Acousmatics as disciples of Pythagoras who, for five years, listened to the master speak from behind a veil, observing the strictest silence. Schaeffer metaphorized the analogue medium into that veil, to unveil a hearing to which we have grown accustomed: listening — on the telephone, tape, the radio — to sounds whose original source remains unseen.

The musical note, a glaring assortment of pitch, duration, and intensity, has borne sway over European tradition and laid claim to universality. Owing to a notational system, the composer sings in silence, plays in silence, sight-reads in silence. The score prefigures the work, which is one and the same with the symbols of writing: 'Beethoven's quartets lie in the storerooms of the publishing house like potatoes in a cellar' (Heidegger 1935). The composer does not hear but reads, 'pre-listens'. Schaeffer likens his demarche to the scholastic exercise of translating a text from one's mother tongue into a foreign language. The performer translates symbols and notions *into sound,* and an implicit, readable work, becomes explicit, audible to laymen. Still, there is something sonorous in a musical

composition. 'The thingly element is so irremovably present in the art work that we are compelled rather to say conversely that the musical composition is in sound' (ibid.). The sound recordist does not read but listens. Comparing the sound image generated by the electroacoustic chain with the original sound phenomenon, which originates from real instruments and unfolds in real magnitude over the acoustic field, he translates *from sound.* Schaeffer likens this demarche to the scholastic exercise of translating a text from a foreign language into one's mother tongue.

In the first century Plutarch expostulated with youth about the exertion of speech to the detriment of listening: to listen extempore is ill-advised! 'He who plays the ball simultaneously learns to throw it and to catch it, but when it comes to speech, on the contrary, reception takes precedence of deliverance, just as conception and pregnancy precede birth.' In 1931 Spengler showed vision as the nordic predator's sense par excellence, hearing as the prey's. In 1953 Barthes traversed the geometry of the writer's space: horizontally, the *language,* a consensual corpus common to all writers of a period; vertically, *style,* a repertoire of gestures — imagery, delivery, vocabulary — springing from the writer's past; obliquely, *écriture,* an act of historical solidarity binding the writer's utterance to the vast History of the Others. Schaeffer set sail from Literature in avoidance of the commitment of *écriture* for which writing pleaded. Bound for music, his commitment was all too clear: to reconcile technology to nature. Substituting perception for expression as the locus of such a commitment, he raised *écriture* to the second power; substituting hearing for seeing, he cubed it. Notwithstanding, three fallacies — 'Schaeffer is a composer', *'écriture* is writing', and 'written note and written word are the selfsame sign' — have compacted into a petitio principii: 'Schaeffer is the antithet to the *écriture* composer'.

When a boy communicates he collects his thoughts; he makes silence. He awaits something from Himself or his Visitor. He waits upon some Thing to increase the feeling of His presence to It and of Its presence to Him. 'Bereft of words, prior to any intention, adoration is attention, a summons to consciousness' (Schaeffer 1969). Listening reducedly, we receive a sonic thing whose image starts forming in our consciousness. The flow of distanceless uniformity where all things are carried away and mixed up is halted thereby. 'That the thingness of the thing is particularly difficult to express and only seldom expressible is infallibly documented by the history of its interpretation' (Heidegger 1935): a bearer of traits (i.e. the sonic object as qualified by the seven morphological criteria); the unity of a manifold of sensations (i.e. the transcendental sonic object); formed matter (i.e. the shape/matter pair that underpins morphology). That remoteness, however near, is the sonic thing itself. From the objectlessness of the standing reserve Schaeffer elicits a sonic object that relapses into there as musicalness reservoir. There is in the sonic object 'the impetus of a break and the impetus of a coming to power, there is the very shape of every revolutionary situation, the fundamental ambiguity of which is that Revolution must of necessity borrow, from what it wants to destroy, the very image of what it wants to possess' (Barthes 1953). For all that, the sonic object is not an aesthetic product but a signifying

practice, not a structure but a structuration, not an object but a work and a game, not a group of closed signs but a volume of traces in displacement, not signification but the signifier, not the old musical work but the Text of Life (cf. Barthes 1974).[9]

In the fourth century B.C. Lieh-tzu asseverated: 'The perfect discourse is wordless, the perfect act is not to act.' In 1857 Baudelaire limned the Orient of the Occident, the China of Europe, an Occidental China, where Nature was recast by dream. In 1881 Nietzsche argued: 'You say that food-place-air-society determine and transform you? All the more so do your opinions, for they determine you as to your choice of food-place-air-society.' In 1925 Artaud avowed: 'I suffer from the translation Mind, the Mind that intimidates things so as to make them enter into the Mind.' In 1974 Barthes uphold: 'We must above all aim at *fissuring* the meaning system itself: we must emerge from the Occidental enclosure'. In 1995 Handelman averred: 'To alter our way of perceiving is to alter who we are, to alter the structure of knowledge and the process of knowing itself. Similarly, alterations in knowledge and the process of knowing are impossible without corresponding changes in perception.'

With reduced listening, Schaeffer has brought hearkening to sonic things up to date with the poetics of Varèse, Scelsi, Ponge, Freud, Heidegger, Barthes, Lacan, and Calvino. With the sonic object, he has brought listening to recorded sounds into the world of significations. With acousmatic listening, he has brought the tape machine into play as a component of a global rebirth of culture. With musique concrète, he has brought the technical object's *concretude* (see Simondon 1958) to bear upon the 'problematics' of Western composition. Schaeffer died on 19 August 1995: 'My essential role is to communicate a manner of comprehending, feeling, and acting that may seem, from the outside, terribly personal. In fact, I am but a relay myself.'

Artaud, A. (1925) *L'ombilic des limbes.* Paris: Gallimard.

Barthes, R. (1953) *Le degré zéro de l'écriture.* In *Œuvres complètes* I. Paris: Seuil, 1993.

—— (1974) L'aventure sémiologique. *L'aventure sémiologique*. In *Œuvres complètes* III. Paris: Seuil, 1995.

—— (1977) Écoute. In *Œuvres complètes* III. Paris: Seuil, 1995.

Baudelaire, C. (1857) L'invitation au voyage. *Le spleen de Paris* XVIII. In *Œuvres complètes* I. Paris: Gallimard, 1975.

Bayle, F. ed. (1980) *Répertoire acousmatique 1948–1980.* Paris: INA-GRM.

Benjamin, W. (1936) *Das Kunstwerk im Zeitalter seiner technischen Reproduzierbarkeit.* In *Gesammelte Schriften* I.2. Frankfurt am Main: Suhrkamp, 1990.

Brunet, S. (1969) *Pierre Schaeffer.* Paris: Richard-Masse.

[9] In 1975 Stefani opened the proceedings of the First International Congress on Musical Semiotics avouching that the object of musical semiotics was the score, since no theory of the musical Text existed yet, and Schaeffer's sonic object would not be reckoned with.

Brunet, S. ed. (1977) *Revue musicale* 303–5.

Cadoz, C. et alia (1984) Responsive Input Devices and Sound Synthesis by Simulation of Instrumental Mechanisms: the Cordis System. *Computer Music Journal* 8 (3).

Chion, M. (1980) *Pierre Henry.* Paris: Fayard/SACEM.

Handelman, E. (1995) Cybersemiosis: 2nd Worlds — 2nd Perception. http://www.music.princeton.edu/~eliot/cybersem.1html

Hart, J. (1989) Préface. In Gilbert Simondon 1958.

Heidegger, M (1935–6) Der Ursprung des Kunstwerkes. *Holzwege.* In *Gesamtausgabe* V. Frankfurt am Main: Klostermann, 1977.

—— (1954) Die Frage nach der Technik. *Vorträge und Aufsätze.* Pfullingen: Günther Neske.

—— (1954) Das Ding. *Vorträge und Aufsätze.* Pfullingen: Günther Neske.

Hofstadter, A. (1971) Introduction. In Martin Heidegger 1971, *Poetry, Language, Thought.* New York: Harper and Row.

Kirk, G.S. and Raven, J.E. (1957) *The Presocratic Philosophers: a Critical History with a Selection of Texts.* Cambridge: Cambridge University Press.

Lerdhal, F. (1987) Timbral Hierarchies. *Contemporary Music Review* 2 (1).

Lieh-tzu (732) *Le Vrai Classique du vide parfait.* In *Philosophes taoïstes.* Paris: Gallimard, 1980.

Littré, P.-É. (1877) *Dictionnaire de la langue française.* Chicago: Encyclopaedia Britannica, 1978, 4 vv.

Machover, T. ed. (1984) *Contemporary Music Review* 1 (1).

Mitcham, C. (1994) *Thinking Through Technology: The Path between Engineering and Philosophy.* Chicago: University of Chicago Press.

Nietzsche, F.-W. (1881) Frühjahr — Herbst 1881. *Nachgelassene Fragmente* 1880–1882. In *Sämtliche Werke: kritische Studienausgabe herausgegeben von Giorgio Colli und Mazzino Montinari* IX. Berlin: de Gruyter, 1988.

Palombini, C. (1993) Pierre Schaeffer, 1953: towards an Experimental Music. *Music & Letters* 74 (4): 542–57.

—— (1997) Technology & Pierre Schaeffer. *MikroPolyphonie* 4.01. Melbourne: La Trobe University, http://farben.latrobe.edu.au/mikropol/volume4/palombini-c/palombini.html

Plutarch (n.d.) *L'arte di ascoltare.* Milano: Modadori, 1995.

Risset, J.-C. (1966) *Computer Study of Trumpet Tones.* Murray Hill: Bell Laboratories.

Schaeffer, P. (1938) *Vingt leçons et travaux pratiques destinés aux musiciens mélangeurs.* Paris: Radio Nationale (internal publication).

—— (1941) Esthétique et technique des arts-relais. In Sophie Brunet ed. 1977.

—— (1941) Technique et esthétique des arts-relais. In Marc Pierret 1969, *Entretiens avec Pierre Schaeffer.* Paris: Pierre Belfond.

—— (1946) L'élément non visuel au cinéma: (I) Analyse de la bande 'son', (II) Conception de la musique, (III) Psychologie du rapport vision-audition. *Revue du*

cinema 1 (1–3): 45–8, 62–5, 51–4.
—— (1946) Notes sur l'expression radiophonique. *Machines à communiquer: I. Genèse des simulacres.* Paris: Seuil, 1970.
—— (1950) Introduction à la musique concrète. *Polyphonie* 6: 30–52.
—— (1952) *À la recherche d'une musique concrète.* Paris: Seuil.
—— (1953) Vers une musique expérimentale. In Pierre Schaeffer ed. 1957, *Revue musicale* 236: 11–27.
—— (1957) Lettre à Albert Richard. In Pierre Schaeffer ed. 1957, *Revue musicale* 236: iii–xvi.
—— (1960) Note on Time Relationships. *Gravesaner Blätter* 17: 41–77.
—— (1966) *Traité des objets musicaux: essai interdisciplines.* Paris: Seuil, 1977 (enlarged reprint).
—— (1969) *Réflexions de Pierre Schaeffer.* In Sophie Brunet 1969.
Schaeffer, P. and Mâche, F.-B. eds. (1959) *La revue musicale* 244.
Schafer, M. (1977) *The Tuning of the World.* New York: Knopf.
Simondon, G. (1958) *Du mode d'existence des objets techniques.* Paris: Aubier, 1989 (iii).
Smalley, D. (1986) Spectro-morphology and Structuring Processes. In Simon Emmerson ed. 1986, *The Language of Electroacoustic Music.* London: Macmillan.
Spengler, O. (1931) *Der Mensch und die Technik: Beitrage zu einer Philosophie des Lebens.* München: Beck'sche.
Stefani, G. ed. (1975) *Actes du Premier Congrès Internacional de Sémiotique Musicale.* Pesaro: Centro di Iniziativa Culturale.

Música Fractal: As novas tecnologias da musica contemporânea

Frederico Richter (Frerídio)*
Universidade Federal de Santa Maria
Prédio 40 (CAL), 1° Andar, s. 1211
Campus da UFSM, Faixa de Camobi, Km 09 -
97119-900 – Santa Maria - RS - Brasil

Resumo

Este artigo trata da aplicação de sons fractais na composição musical e a compatibização de estruturas novas com as já existentes. Apresenta, ainda, uma análise de obras fractais e os caminhos seguidos por elas.

1 - Os Fractais: uma análise estética e histórico-científica

Música e cálculo sempre andaram juntos desde a antigüidade. A teoria musical, a composição e a construção de instrumentos derivam de leis da física e da matemática, segundo David Ernst(1977). Béla Bartok usou a Seção Áurea para a *gestalt* de suas obras (*gestalt* entendida como forma e conteúdo interno- sugerimos que seja usada a palavra **contextura**, em lingua portuguesa). A tendência de usar a ciência para estruturar criações, sejam de música ou das artes em geral, sempre existiu. Benjamin Boretz e Edward T. Cone (1971) escreveram que não se pode ignorar os compositores que pesquisaram e ainda pesquisam nos Estados Unidos, tais como Milton Babbit, J.K. Randal e Arthur Berger. Todos estudaram exaustivamente os sistemas de 12 sons e especialmente as obras de Schöenberg onde a aplicação numeral é marcante e foi estendida por estes compositores para cálculos mais complexos. Os mesmos autores dizem que, durante os últimos 50 anos, houve várias tentativas, na teoria musical, cientificamente orientadas mostrando o caminho do futuro ao compositor e ajudando-o a encontrar uma base firme no período de caos que seguiu à desintegração da escala maior e menor (sistema tonal) no começo deste século. Esses autores falam da existência de um movimento para Quarteto de Cordas de Joseph Haydn, composto em seu comprimento (duração) conforme as proporções da Seção Áurea.
Neste século, novos meios emergiram e novas estéticas advindas da Ciência (física), Tecnologia (computador e informática) e da Matemática (proporções e domínio racional) surgiram. Entre elas, as teorias da não linearidade: em conseqüência, a física não linear em contraposição à física tradicional, a Geometria Fractal em contraposição à Geometria Euclidiana e a Tecnologia de Ponta em contraposição à Tecnologia Mecanicista. Disto tudo, o mais importante foi o desenvolvimento do computador.
Este acelerando científico influenciou todas as manifestações culturais do homem. E a Música Fractal também necessita para seus cálculos recursivos de velocidade, portanto do computador (gráfico). Há dois aspetos a considerar a respeito dos fractais:

1° aspecto: o científico, o da matemática, da física não linear e campos correlatos (interdisciplinares).
2° aspecto: o estético, escondido nas formulações teóricas e que nos é revelado pelo computador gráfico - este é o aspecto que pode e deve ser empregado e utilizado pelas artes.
Quando analisamos nas artes o 1° aspecto, o científico, vemos que as artes em todos os tempos estiveram atreladas à física e a cálculos: na acústica e nas cores, à física; nas tintas, à química etc.. As ciências exatas sempre serviram as artes com standars de precisão. A Geometria Euclidiana, foi empregada por pintores e artistas plásticos para limitar o espaço em suas criações (círculos, quadrados e outras formas geométricas artificiais que praticamente não existem na natureza).
Na Geometria Fractal, denominada por Benoit Mandelbrot (1983) de Geometria da Natureza e que tem uma grande beleza, temos a medição das coisas da natureza, as plantas, as nuvens, as costas dos mares e rios etc.. É a descoberta e a proclamação de uma nova estética (Peitgen, The Beauty of Fractals, 1986). Suas qualidades são a auto-similariedade, a medição do grande e do pequeno, a expressão do caótico ou do **Caos** que tem uma ordem que se expressa em "ordem-desordem-ordem-desordem" o que é, em si, uma ordem .
Não somos matematicos nem cientistas, somos artistas e compositores. E isto é importante de ser dito, de vez que **não temos todas as respostas** e provavelmente nunca as teremos todas juntas. Não manejamos os cálculos precisamente , deixamo-los à vontade - mais musicalmente, "ad libitum" - de conformidade com o que acontece com líquidos que, acelerados, têm vontade própria e imprevisibilidade, conforme foi demonstrado pelo físico Lorenz em seus estudos. **Imprevisibilidade** e **Chance** vem sendo pesquisados desde Yannis Xenakis e pela música aleatória. O que fazemos: respeitamos esta imprevisibilidade, tiramos proveito dela e, acima de tudo, seguimos a grande linha composicional expansiva do passado para desvendar o futuro. O presente são as obras que ficam.

2 - Música fractal - o manejo dos novos meios expressivos

A música fractal veio ao nosso encontro através dos livros de Mandelbrot. Ficamos fascinados com o mundo novo que se abria. E, começamos a estudar e ler a respeito. Assim, chegamos ao pragmatismo de realizar algo em música, estudos, pesquisas, música experimental e obras que consideramos definitivas, dentro dos meios expressivos novos e caóticos.
Dentre os artigos sobre música fractal que lemos, chamou-nos a atenção a maneira como os pesquisadores encaram esta música fractal. Para discutir isto, e descrever o que fazemos, queremos analisar um destes artigos. Este artigo é um referencial, dentre outros, porque o autor foi diretamente influenciado por Benoit Mandelbrot. Foi publicado pelo "Computer Music Journal" e se intitula PROFILE: A MUSICAL FRACTAL de Charles Dodge do Brooklyn College da City University of New York (1988). Após uma breve introdução, o autor propõe dois aspectos para o uso de algoritmos: o primeiro é um programa que deve dar parâmetros da música e o segundo, um método para qualificar fractais, tias como

"Wiggly", "Hydralike" e "Wrinkled"; são a essência do comportamento caótico. Está descrito em diversos escritos de físicos que tratam o assunto, dentre eles o próprio Mandelbrot. Faz, ainda, considerações sobre formas fractais, estruturas amorfas etc..
"Floco de Neve" de Koch é usado por Dodge para estruturar sua obra Profile, como ponto inicial. Parte do *1/f noise* (ruido 1/f) algoritmo citando os *white noise* (ruido branco) e *brownian noise* (ruido browniano). A auto-similaridade na constituição destes ruidos, em sua densidade, o atrai - após, começa a descrever uma coleção cromática de notas. O número exato de notas é determinado estocasticamente pelo compositor em pitch-classes dentro das quais gira o aleatório onde o uníssono e a oitava são previamente encorajados. O autor cria uma série de sons repetidos ou não, que são mostrados. Em Profile ele imita a forma dos flocos de neve de Koch através de 3 linhas musicais. São realizados cálculos precisos de multiplicação por um tempo constante etc.. O artigo detalha como o autor "civiliza" o caótico para construir uma peça que, a nosso ver, dificilmente terá as características de um comportamento caótico. É um bom método para fazer música nova, nossa dúvida está em designá-la caótica após tantos preparos, métodos e procedimentos manipulatórios afetando os parâmetros básicos do caótico. Por definição, o caótico é imprevisível e como tal deveria ser tratado. Se tirarmos dele esta qualidade, a auto-similaridade de *per si* não é suficiente para caracterizar o fractal. Uma constante produz auto-similaridade sempre. Esta abordagem do autor parece-nos um contraponto com regras novas e primitivas como no início do contraponto tradicional que depois se abriu num leque espiralado através dos séculos, cada vez mais complexo. A Ars Nova já tinha seu próprio aleatório através da *iso-ritmia.* Pensamos que o caótico não pode ser freiado: se freiado torna-se não caótico, normal. Sabemos que existem fractais determinísticos. Estes não nos interessam, e não cremos que sejam úteis para uma aplicação nova da tecnologia fractal em música. O uso destes fractais deterministas pode produzir equívocos. O novo deve ser inequívoco.
Do mesmo modo, consideramos que a composição musica fractal não se diferencia dos outros métodos composicionais do passado. A grande diferença está nos novos meios utilizados e, estes sim, devem ser respeitados em seus parâmetros fundamentais: impredictabilidade, auto-similaridade e não-linearidade.Para isto devemos evitar os fractais deterministas.

3 - Procedimentos composicionais - a estética

Desde os inícios de nosso século vinte já a findar, o Futurismo, com todos os seus multi-aspetos, influenciou o mundo para gerar uma nova ordem estética, moral e ética. Usou para isto, o caos, a anarquia,a propaganda, a agressividade, a politica, a guerra, a moda e todos meios que poude inclusive,o fascismo. As *antinomias* estavam no ar, elas queriam e precisavam de seu espaço para o que se julgava necessário para a continuidade e progresso da arte.
As antinomias são procedimentos contrários - se isto é moralmente belo, o contrário seria o indicado de agora em diante. Nas artes aconteceu o mesmo. Na música, surge a "orquestra"

ideal: o modêlo seria o *intonarumori* de Russollo e a orquestra seria de ruidos, tosses, espirros, esplosões etc.. Os Manifestos Técnicos mostram que, além da política, se queria usar também as ciências para justificar as *antitudo*. Na composição musical, em sua criação, tentou-se introduzir uma estética agressiva e contrária a tudo o que se fazia antes. Esta atitude teve, na verdade, alguns frutos bons e muito frutos máus.
Simplesmente fazer o contrário de uma coisa já existente, é *plagiar e não criar o novo*. Pode ser um critério, mas, nos parece muito simplista. A contininuidade da música não exigiu tal radicalismo.
Futurismo, com seus manifestos técnicos e sua crua insensibilidade queria justificativas na ciência para uma estética fria, calculista e, naturalmente nova. A ciência não se intimidou. Deu uma nova estética para o mundo, um novo conceito de beleza baseado em cálculos e usou o caos (usado pelo Futurismo realistica e equivocadamente, fazendo o que, vulgarmente, chamamos mesmo de caos), de uma maneira científica, nova e até o justificou. Esta ciência, que sempre esteve presente nas artes, também desta vez, não se omitiu: revelou-nos os FRACTAIS e suas belezas incríveis. Isto gerou uma nova estética bem superior àquela estética gerada por "ideais" bastante questionáveis do chamado Futurismo Italiano. O século XX, iniciado por guerras, parece que vai findar em paz.

4 - *Monumenta Fractalis -Thomas* : detalhamento na construção desta obra musical.

Esta obra foi escrita em 1991 para órgão de tubos e fita magnética. É uma homenagem a Thomas Tallis, compositor inglês do século XVI e conecta o estilo fractal ao estilo gótico-flamengo de Tallis. Utilizamos programas em **Basic** para gerar as estruturas fractais. As manipulações destas estruturas sonoras foram compatíveis com os parâmetros caóticos e não-lineares e se destinaram somente para diminuir a velocidade com que as estruturas surgem, pois, é difícil fazer música com esses sons fugidios e rápidos. É possível diminuir a velocidade mudando números nos algoritmos. E isto não altera as estruturas dos fractais. Apenas aumenta sua duração de forma inversamente proporcional: mais velocidade, diminue a duração, menos velocidade, aumenta a duração.
Procuramos analogias estruturais entre o fazer composicional do passado e as manifestações sonoras dos algoritmos e descobrimos, por exemplo, traços de iso-ritmias na música de Thomas Tallis, que deram unidade ao seu pensamento musical. Por outro lado, o cadenciamento apresentado no Coral - "Se me amas, guarda os meus mandamentos" foi tranferido para as estruturas fractais, surgindo assim, um cadenciamento novo, às vezes, com uma nota só, repetida, propiciando-nos transmitir algo coerente dentro da estruturação composicional universal. Ainda no terreno das analogias com o Coral de Thomas Tallis, fizemos um coral com vozes fractais. Neste momento a música fractal aparece sozinha, dando o clima do novo.
Para finalizar a peça, o órgão faz comentários musicais sobre o conteúdo da fita magnética (fractal) : mais um vínculo entre as duas estéticas. Assim, procuramos unir o século XVI ao

século XX, quasi XXI. Para o ouvinte, embora a distância de séculos, fica a coerência na semelhança das estruturas.

5 - Procedimento construtivos

Usamos um Programa Basic para que o computador nos desse resultados em freqüencias musicais. Em *natura,* aparentemente, os fractais musicais não são aproveitáveis. Como já dissemos, precisamos manipulá-los para que o ouvido humano possa "digerí-los", isto é, possa assimilá-los, levá-los à nossa mente para serem compreendidos. O belo, o estético, precisa de uma conpreensão superior da mente. E esta compreensão é uma das funções superiores do homem. Nosso procedimento em relação aos sons fractais foi, torná-los acessíveis para a compreensão auditiva. A primeira coisa para isto, é **diminuir a velocidade de sua aparição no tempo.** Com isto se consegue penetrar na construção e na arquitetura destes sons rapidíssimos. E mais material em sua extensão, porque os sons são mais longos.O resultado é que podemos usar menos material real. O que é bom, pois, traz em si, uma regra fundamental de construção: **a economia de meios.**
passo seguinte: uma análise técnica do material e suas possibilidades de aproveitamento em uma obra de arte. Para isto foi necessária, uma análise estética. Nós queríamos conjugar, numa simbiose, meios tradicionais, com meios não tradicionais.Isto não significa arte tradicional, apenas usamos técnicas já conhecidas, não necessariamente tradicionais. Unimos o órgão ao computador, ou melhor, aos seus resultados. Resultados com resultados, precisou-se do uso e da inventabilidade de novos meios para esta anunciada simbiose. Aí, entram as **analogias estruturais entre o fazer composicional do passado** e, os novos meios que usamos. Unir a trilha sonora fractal com a trilha sonora do órgão, a primeira numa fita magnética e a segunda em partitura a ser recriada pelo organista foi uma tarefa de criação extremamente complexa. Usamos técnicas de minutagem para sincronizar as duas fontes sonoras o que exige precisão. Usamos as analogias estruturais que compreendem forma e conteúdo. Precisamos de três gravadores e um mixer para realizar a trilha fractal. As fontes fractais foram diversas. Tínhamos trilhas com diversos **modos**: classificamos estes modos em modos mais calmos, mais turbulentos, mais lentos, mais rápidos, tudo subordinado à nossa visão estética em face da trilha do órgão já feita em estilo próprio, impregnado do estilo de Thomas Tallis. Dentro das analogias, construimos a obra como se constroi uma obra em todos os tempos. Por exemplo, as cadências: é quando a obra respira e o ouvido "digere" o que ouve. As pausas tem a mesma função. Assim, construimos esta obra, como se faz em composição musical. O ouvido nos foi um guia para evitar repetições desnecessárias, acúmulo de sons que poderiam desagradar. Afinal, o bom gosto deve estar presente em qualquer obra de arte. O executante realiza a sua partitura ouvindo a fita magnética com a trilha fractal. Introduzimos "compassos de suspensão", como fermatas. Possibilitam ao organista a eventual espera e/ou resincronização com a fita.

6 - Referências bibliográficas

BERNARDINI, Aurora Fornoni. Org. O Futurismo Italiano. São Paulo: Editora Perspectiva, 1980.

BORETZ, B. e CONE, E.T. Perspectives on American Composers.New York: W.W. Norton, 1971.

COPE, David. New Music Composition. New York: Macmillan Publishing, 1977.

DODGE, Charles. Profile: A Musical Fractal. In: Computer Music Journal. Vol.12, nº3, Fall 1988, pp10-14.

ERNST, David. The Evolution of Electronic Music. New York: Schirmer Books, 1977.

GLEICK, James, Caos-A Criação de uma Nova Ciência, Editora Canpus, Rio de Janeiro, 1988.

MANDELBROT, Benoit. The Fractal Geometry of Nature. New York: Freeman W.H. Company, 1983.

PEITGEN, H.-O. e RICHTER, P.H. The Beauty of Fractals. Berlin: Springer Verlag, 1986.

PEITGEN, H.-O. e SAUTE, D. Editors. The Science of Fractal Images. New York: Springer Verlag, 1988.

SCHAEFFER, Pierre. Tratado dos Objetos Musicais. Brasília: Editora Universidade de Brasília, 1993.

SCHOENBERG,Arnold. Fundamentals of Musical Composition. London: Gerald Strang & Leonard Stein-Faber and Faber, 1977.

SCHWENK, Theodor. Das Sensible Chaos. GmbH Stuttgart: Verlag Freies Geistesleben, 1962.

TISDALL,Caroline e BOZZOLA, Angelo.Futurism. Singapura: Thames and Hudson, 1996.

Uma Representação de Conhecimento em Harmonia Musical

LUCIÊNIO DE MACÊDO TEIXEIRA
Departamento de Artes / Laboratório de Informática Aplicada às Artes – CH-UFPB
lucienio@liaa.ch.ufpb.br

EDILSON FERNEDA
Departamento de Sistemas e Computação – CCT-UFPB
edilson@dsc.ufpb.br

EVANDRO DE BARROS COSTA
Departamento de Informática e Matemática Aplicada – CCEN-UFAL
ebc@fapeal.br

CCT/UFPB – Caixa Postal 10.090
58.109-970 Campina Grande, PB

Abstract: The lack of knowledge representation in the domain of teaching musical harmony is the main motivation for the present paper. As a strategy, in the paper we begin by evaluating that material traditionaly used in teaching Harmony such as specific exercices, formal concepts, etc with the aim of enlarging the common representation of concepts, objects, and structures found in this domain. Some considerations on the formal knowledge representation are discussed in the paper. Finally, a general structure for musical concepts or objetcts is presented including their relationships

1 Introdução

Na busca de ferramentas para o suporte às atividades ligadas ao domínio da Música, diversos sistemas computacionais têm sido desenvolvidos, notadamente aplicações voltadas para a Teoria Musical, tais como sistemas de notação musical, reconhecimento de partituras, sistemas de análise, sistemas tutoriais e sistemas de composição.

Nesta perspectiva, o projeto no qual se insere este trabalho visa o desenvolvi-

mento de um Sistema Tutor Inteligente no domínio da Harmonia Musical (Teixeira 1997). Para isso, investiu-se inicialmente nos aspectos de organização e representação do conhecimento deste domínio.

Aqui, é proposto um modelo de representação e manipulação do conhecimento musical que permita a sua utilização de forma coerente em um ambiente computacional, voltado ao ensino de Harmonia Musical.

Como estratégia, o material tradicionalmente utilizado no ensino de Harmonia (exercícios específicos, conceitos formais, etc) foi avaliado com o intuito de dar uma maior abrangência no que se refere à representação dos conceitos, objetos e estruturas encontradas neste domínio.

2 Algumas considerações sobre representação de objetos musicais

Para demonstrar a complexidade e a estruturação da Música, mais particularmente da Harmonia, basta analisar seus componentes. Percebe-se logo de início que os objetos musicais estão posto em dois grupos distintos, *formal* e *cognitivo* (Dannenberg 1993), que respectivamente são:

- os objetos que podem ser tratados matematicamente, que em geral são os que podem ser dispostos em uma partitura, por exemplo: notas, tempo, harmonia, alterações, tonalidades, freqüências, etc e
- os objetos que se encontram no domínio do perceptivo. Aqui cabem todas as informações que, por sua natureza, não podem ser organizadas em uma folha de papel. Pode-se citar como exemplos as questões de tensão e relaxamento, expectativa, emoção, contexto histórico e performance.

Estes dois grupos interatuam de forma a se complementarem (Ferneda, Silva, Teixeira & Silva 1994). Mesmo quando das tentativas de representação da parte formal, os pesquisadores sempre buscam, na medida do possível, levar em consideração os aspectos, em geral estudados em Musicologia, Análise e Composição, que não "aparecem" explicitamente na partitura. Separar uma coisa da outra, genericamente, pode terminar por resultar em uma "caricatura" do domínio alvo.

Como conseqüência deste relacionamento, pode-se presumir os mais variados níveis de representação que podem ser encontrados em uma obra musical. Isso se dá pela forma como necessita ser direcionada uma representação, a depender do objetivo (relevância), onde alguns aspectos devem ser colocados em segundo plano e/ou ignorados (abstração) (Wiggins, Miranda, Smaill & Harris 1993).

Embora este trabalho contemple os objetos do primeiro grupo, existe a preocupação em tornar disponível esta representação de forma a permitir seu uso em estágios mais avançados do ensino de Harmonia. Aqui é cabido dotar a representação de mecanismos que façam referência a diversos contextos (Teixeira, Costa, Silva & Ferneda 1995).

A conclusão é que, embora se trabalhe com determinadas partes formais do universo musical, há a necessidade de não negligenciar a existência de todo um domínio informal (cognitivo) que é de suma importância na resolução de determinados problemas (Widmer 1992).

3. Representação de estruturas musicais

Após a separação dos elementos desses dois grupos (formal e cognitivo), o passo seguinte é definir a *estruturação* e *hierarquização* da representação. Por *estruturação* entende-se a organização do material sonoro (de acordo com a literatura básicade ensino de Harmonia (Hindemith 1986, Kostka 1984, Schoenberg 1979)), de maneira a compor estruturas que encerrem em si mesmas um significado completo (Dannenberg 1993). Estruturas menores podem continuar gerando outras estruturas, sempre respeitando um sentido estético e formal. Já a *hierarquização* é a disposição lógica dessas estruturas onde pode haver dependências, abstrações ou heranças entre elas.

Um outro cuidado a ser tomado é resultado da aparente simplicidade da informação musical. Esta pode levar à criação de estruturas aparentemente não tão complexas e por isso deve-se esquematizá-las de maneira a permitir que as mesmas possam ser estendidas posteriormente. O conceito de tonalidade, por exemplo, pode ser visto de forma equivocadamente simples (como a definição de uma escala por exemplo). Por outro lado, a definição de tonalidade implica na definição de uma nota inicial, um conjunto de intervalos predefinidos, um coleção fechada de acordes e uma escala também formada por intervalos predefinidos. Dependendo do ponto de vista, pode-se restringir inconscientemente uma estrutura, deixando sua representação aquém das necessidades reais.

Como passo inicial, deve-se definir as primitivas e como estas devem ser estruturadas. Já aqui, um descrição puramente teórica pode produzir as "caricaturas" anteriormente referidas. Portanto, o conhecimento cognitivo deve servir de *interprete* do formalismo utilizado, na intenção de definir o que é adequado ou não, ou se o potencial teórico está bem aplicado.

É justamente neste momento que se faz a separação nos dois grupos descritos no tópico anterior (representação de natureza teórica e representação de natureza cognitiva). Estas representações irão assumir diferentes perspectivas, de acordo com um domínio musical específico (Honing 1993). Neste ponto, deve-se compreender

que dificilmente existirá uma representação ou conjunto de representações que consiga envolver todas as esferas de ação da Música. Para cada domínio devem ser considerados diferentes aspectos e, a depender das ações, a parte formal pode ser preponderante em relação à parte cognitiva e vice-versa.

Entre as áreas do conhecimento musical (mais especificamente em Análise e Produção Musical) onde a representação formal (teoria), segundo Honing (Honing 1993), prepondera em relação à representação cognitiva estão a *Musicologia*, a *Música Computadorizada*, a *Edição Musical* e os *Sistemas de Recuperação*.

4 Uma proposta de representação

O que se deseja com a representação aqui considerada, é a sua utilização voltada ao ensino de Harmonia. A principal preocupação desta representação é a de prover mecanismos que possam suprir as necessidades de ensino e aprendizagem em Harmonia. Nesta fase do trabalho preocupou-se apenas em representar as estruturas básicas envolvidas no ensino de Harmonia propriamente dito. Todas as estruturas mais complexas da Harmonia são, de uma maneira ou de outra, variações dessas estruturas básicas.

A Harmonia é um exemplo típico de domínio, onde se constata a presença de objetos estruturados. Por este motivo, a primeira providência foi identificar quais elementos musicais são preponderantes na resolução de problemas harmônicos. Nesta primeira fase, a preocupação foi a de isolar as menores partículas musicais e, partindo delas, ir construindo os objetos subseqüentes por ordem de complexidade.

Tomando a definição do objeto *nota* como base das estruturas musicais, tem-se como resultado o fluxograma apresentado na Figura 1, com a dimensão das relações entre os vários objetos desse domínio. Aqui a hierarquia intrínseca vai do objeto mais simples (estruturalmente) até o mais complexo.

Seguindo esta ordem encontra-se:

- que os Intervalos estão em relação de dependência e de herança horizontal* com o objeto Notas;
- que as Escalas e Acordes estão em relação de dependência e herança horizontal com o objeto Intervalo e que de acordo com sua disposição (vertical ou horizontal) gera um ou outro,

* Entende-se por *herança horizontal* a relação onde o valor de um atributo de um objeto é definido pelo valor deste mesmo atributo em um dos componentes do objeto. Por exemplo, a definição de um *Intervalo* pressupõe a definição de duas notas e seus atributos intrínsecos e, conseqüentemente, o objeto *Intervalo* herda horizontalmente os atributos de *Notas*.

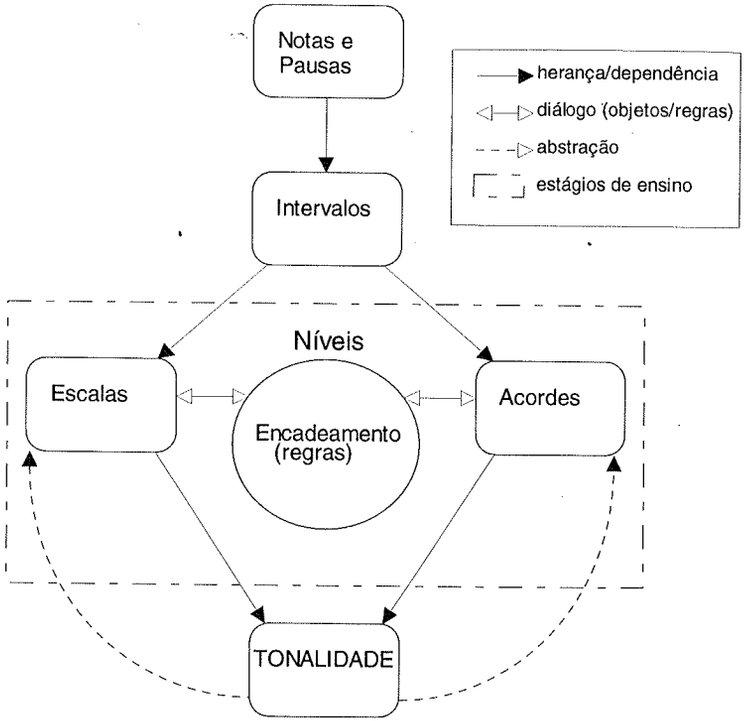


Figura 1: estruturação dos objetos musicais relevantes à Harmonia

- que alguns atributos e métodos de Escalas e Acordes são definidos pelo objeto Níveis, que representa os vários estágios de aprendizagem,
- que a relação entre Escalas e Acordes com o objeto Encadeamento se dá através de métodos (regras) de manipulação e tratamento;
- que Escalas e Acordes definem o objeto Tonalidade ao mesmo tempo que são uma abstração desta última e
- que Tonalidade define um conjunto singular de Acordes e uma única Escala.

De maneira similar, e principalmente em relação aos acordes, talvez o objeto de maior interesse em Harmonia, a sua estrutura deve permitir certas possibilidades de inferência, tais como:

- a partir de um acorde incompleto chegar às notas faltantes,
- de um acorde complexo deduzir sua construção básica (sem dissonâncias, etc),

- construir todas as possibilidades de inversões de um acorde,
- localizar, dentro de uma tonalidade, qual a função de determinado acorde.

Torna-se evidente que para cada um dos exemplos acima, quando tratados por uma representação orientada a objetos, cada "metamorfose", ou variação, deve ser vista como objetos independentes. Esta preocupação é necessária pelo fato de que tal complexidade deve ser encarada com um certo desvelo, não só pelo motivo da representação em si, mas, principalmente, pelas necessidades inerentes ao ensino e aprendizagem de tal matéria, como dito anteriormente.

Ainda dentro do objeto *Níveis*, tem-se o objeto *Encadeamento*. Este é responsável pelas regras que irão guiar as diversas possibilidades de resolução de um encadeamento. Mais uma vez os mesmos níveis de aprendizado impostos a um aprendiz humano devem ser ressaltados: neste caso, determinadas regras podem ser relevantes em um instante e, em outro, podem ser relegadas a um segundo plano.

Algumas considerações sobre o objeto *Tonalidade*: a *Tonalidade*, em um determinado instante do ensino de Harmonia, possui uma função básica: definir quais acordes e quais escalas farão parte de um problema. Isso possibilita a eliminação de uma grande parte do conjunto de acordes e escalas quando da resolução de um exercício.

As *cifras* estão associadas aos *acordes* e são responsáveis por determinar suas funções. Dessa maneira, uma *cifra* pode estar associada a diversos *acordes*, sendo que a definição biunívoca entre *acordes* e *cifras* é especificada pela *tonalidade*.

Algumas considerações sobre o objeto *encadeamento*. Existem de dois conjuntos distintos de regras que devem ser observadas na resolução de problemas:

- as regras de encadeamento propriamente ditas: são aquelas que regem a maneira como devem se relacionar as seqüências de acordes,
- as regras de distribuição de vozes: responsáveis por manter cada melodia gerada com as regras de encadeamento dentro de parâmetros estéticos aceitáveis (melodias coerentes, dificuldades de execução por um cantor, extensão e respeitar certos relacionamentos entre as demais vozes).

Estes dois conjuntos se inter-relacionam e, com relação à proporção, pode-se dizer que as regras de distribuição são em número menor que as regras de encadeamento. No escopo deste trabalho, não se tem a intenção de representar o universo das regras de encadeamento. O motivo é que, como antes dito, de acordo com o encaminhamento dado pelo professor, pode-se acrescentar ou suprimir um conjunto determinado de regras.

5 Aspectos computacionais da representação

Motivado pela maneira encontrada na organização dos elementos musicais, juntamente com a necessidade de garantir que a manipulação de alguns atributos preponderassem em relação aos demais; optou-se pela utilização do paradigma de *Orientação* a *Objetos* (OO). A consistência do paradigma de OO aplicado à Música pode ser avaliada em diversos trabalhos (Pope 1991). Aqui, os benefícios de modularização, encapsulamento e especificação hierárquica de componentes foram preponderantes na escolha da OO. A conseqüência dessa escolha e de seus benefícios assemelha-se muito a um sistema que tenha como paradigma a utilização de *frames*.

6 Conclusão

Sobre a representação do conhecimento musical realizada, embora não inclua as regras harmônicas (objetivo previsto em trabalhos futuros), pode-se entrever suas contribuições. A capacidade de resolver algumas das questões teóricas de Música, tais como:

- a enarmonia em particular,
- encontrar respostas completas a partir de objetos musicais incompletos,
- capacidade de construir os objetos de escalas, acordes e tonalidades, relevantes ao domínio de Harmonia.

Estas questões podem ser consideradas como incentivo à ampliação deste modelo de representação, com o objetivo de abranger o universo de regras (sejam em contexto Normativo ou Funcional) e de estruturas mais complexas.

Como comprovação deste caminho, logo após a implementação (código) das estruturas relevantes em linguagem LPA-Prolog++ (Vasey 1995), o programa gerado "retornou" à sala de aula para a correção de exercícios aplicados, demonstrando positivamente a sua eficácia.

Atualmente, esforços estão sendo efetuados para o desenvolvimento de um Sistema Tutor em Harmonia Musical tomando como base esta representação e baseada no modelo de arquitetura de Sistemas Tutores Inteligentes MATHEMA (Costa 1997).

Referências bibliográficas

Costa, E.B. (1997). *Um Modelo de Ambiente Interativo de Ensino e Aprendizagem Baseado numa Arquitetura Multi-Agentes*, Tese de Doutorado, COPELE / CCT /

UFPB, Campina Grande, PB.

Dannenberg, R.B. (1993). Music Representation Issues, Techniques, and Systems, *Computer Music Journal*, MIT Press, EUA, 17(3), 20-30.

Ferneda, E., Silva, C.A.P., Teixeira, L.M. & Silva, H.M. (1994). A System for Aiding Discovery in Musical Analysis, *Anais do I Simpósio Brasileiro de Computação e Música*, pp. 177-184, SBC, Caxambu, MG,.

Hindemith, P.(1986). *Harmonia Tradicional*, Irmãos Vitale, São Paulo, 1986.

Honing, H. (1993). Issues in the representation of time and structure in music, *Contemporary Music Review*, 9. (ftp://mars.let.uva.nl/honing/PAPERS/H-93-B.HTML).

Kostka, S., Payne, D.(1984). *Tonal Harmony: with an Introduction to Twentieth-Century Music*, Alfred A. Knopf, New York, EUA.

Pope, S.T. (Ed) (1991). *The Well-Tempered Object: Musical Applications of Objetc-Oriented Software Technology*, MIT Press, EUA.

Schonberg, A.(1979) *Tratado de Armonía*, Real Musical, Madri, Espanha.

Teixeira, L.M., Costa, E.B., Silva, C.A.P. & Ferneda, E. (1995). Aquisição do Conhecimento em Harmonia: Um ambiente de Aprendizagem, *Proceedings of the 2nd Brazilian Symposium on Computer Music*, 290-296, Canela, RS.

Teixeira, L.M. (1997). *Da representação do conhecimento musical ao esboço conceitual de uma sociedade de agentes em Harmonia*, Dissertação de Mestrado, COPIN / CCT / UFPB, Campina Grande, PB.

Vasey P. (1995). *LPA-Prolog++ 2.0, Programmer's Reference*, Logic Programming Associates Ltd., Londres, Inglaterra.

Widmer, G. (1992). Qualitative Perception Modeling and Intelligent Musical Learning, *Computer Music Journal*, 16(2), 51-83, MIT Press, EUA.

Wiggins, G., Miranda, E., Smaill, A. & Harris, M. (1993). A Framework for the Evaluation of Music Representation Systems, *Computer Music Journal*, 17(3), 31-42, MIT Press, EUA.

Sistema Inteligente para a Escolha do Melhor Dedilhado Pianístico

Alexandre B. Viana, José H. F. Cavalcanti
NEUROLAB/COPIN/UFPB
Av. Aprígio Veloso, 882 – Bodocongó
CEP-58109-970 Campina Grande – PB - BRASIL
e-mails: {viana, homero}@dsc.ufpb.br

Pablo J. Alsina
LECA/DEE/UFRN
Passeio dos Girassóis s/n - Lagoa Nova
CEP. 59075-970 Natal – RN - BRASIL
e-mail: pablo@leca.ufrn.br

ABSTRACT

This paper shows the design, implementation details and experimental results obtained from Intelligent System (IS) to aids musical students to learn the optimum melody piano's fingering. The IS uses a Genetic Algorithm for the choice of the optimum fingering.

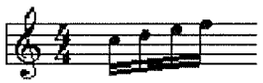
RESUMO

Este trabalho mostra o desenvolvimento, implementação e resultados experimentais obtidos de um Sistema Inteligente (SI) voltado para o ensino de música aos estudantes de piano na obtenção de um dedilhado ótimo de uma melodia musical. O SI utiliza um Algoritmo Genético para a escolha da seqüência do melhor dedilhado.

1. Introdução

Durante a fase de aprendizagem de uma obra musical para piano, o aluno iniciante se sente desmotivado pelo não conhecimento da seqüência dos dedos a serem aplicados nas diferentes teclas do instrumento durante a execução da melodia musical. Por isso é fundamental a presença constante do professor junto ao aluno para que, a partir de exemplos corretos do dedilhado pianístico, ele tenha um melhor aprendizado. Além disso, é importante que o aluno não crie vícios mecânicos no dedilhado. Os vícios mecânicos prejudicam o andamento do dedilhado e o aprendizado da peça musical. Assim, é necessário que o aluno conheça a priori a seqüência do dedilhado a ser usado nos trechos da peça para que sua interpretação seja a melhor possível.

A música é construída por sucessões de sons por grau conjunto (esquema de caráter escalístico) ou graus disjuntos (tipo arpejos ou acordes), ou por uma mistura de ambos. Portanto, o número de combinações possíveis é enorme. Os dedilhados, isto é, as diversas associações de dedos necessárias para executá-los são também muito grandes. Uma seqüência de apenas quatro notas (Figura 1), dependendo do contexto onde está inserida, pode ser digitada, pelo menos de 17 maneiras diferentes (Kaplan 1987).



1 2 3 4
2 3 4 5
3 4 5 1
4 5 1 2
5 1 2 3
2 1 2 3
etc.

Figura 1. Exemplo de dedilhados de uma meldia de 4 notas.

O dedilhado pianístico ótimo, é aquele que o aluno consegue executar o trecho musical da forma mais cômoda possível (sem tensões musculares) e, quando o trecho exigir velocidade de execução, o pianista possa fazê-lo da melhor maneira possível.

Este trabalho está dividido em seções. A segunda seção faz uma breve introdução aos Sistemas Especialistas. Na terceira seção é descrito sucintamente o Algoritmo Genético. Na quarta seção explica-se como o SI encontra o melhor dedilhado, utilizando o SE e o AG. Na seção 5 mostram-se os resultados obtidos para melodias de 5, 6, 7 e 8 notas. Na seção 6 descrevem-se as conclusões e finalmente na seção 7 as referências bibliográficas.

2. Os Sistemas Especialistas

Os Sistemas Especialistas (SE) surgiram na década de 1960 e foram usados na química, geologia, medicina, bancos e seguros (Encarta 1997). Os SE são programas de computadores que tomam decisões ou resolvem problemas usando o conhecimento e as regras definidas por um especialista humano de uma determinada área. Na música, o SE pode, por exemplo, ser usado para auxiliar um aluno no estudo da harmonização de uma linha melódica, usando um estilo musical de uma determinada época da história, ou ajudar a encontrar os melhores dedilhados pianísticos de uma seqüência musical.

A estrutura de um SE assemelha-se à um programa de software convencional. Os principais componentes de um SE são: a base de conhecimento, a máquina de inferência e a interface do usuário.

A Base de Conhecimento pode ser modelada utilizando-se as regras de produção and/or – if/then. Este formato mostra as regras com simplicidade, sem apresentar formas e caracteres codificado para o computador. Por exemplo, supondo-se que o SE usado para o dedilhado pianístico possua a seguinte regra:

SE intervalo entre 1ª nota e 2ª nota for maior do que um intervalo de quinta justa
ENTÃO não usar 1° e 2° dedos.

Observe que deve-se conhecer a melodia, a seqüência de mínimo esforço do dedilhado, etc. Essas informações devem estar de alguma forma armazenado na base de conhecimento.

A Máquina de Inferência utiliza estratégias de raciocínio aproximadas para chegar a uma estimativa válida de uma situação, enquanto possuímos dados incertos e regras imperfeitas.

É de importância fundamental que haja uma Interface amigável e transparente para o usuário. Dessa forma a compreensão do tratamento dado na fase de escolha do dedilhado pianístico, será claramente compreendido.

3. Os Algoritmos Genéticos

A medida do esforço do dedilhado pianístico algumas vezes é um problema NP completo (Sayegh 1989). Diversos pesquisadores (Holland 1992) tem usado algoritmos genéticos para resolver problemas do tipo NP completo.

O Algoritmo Genético (AG) é um procedimento iterativo, que mantém uma população de estruturas (chamadas indivíduos, espécies ou cromossomos) para representarem possíveis soluções de um determinado problema (Grefenstette 1986). Os AG são sistemas que se baseiam nos mecanismos da evolução dos seres vivos. Possíveis soluções de um problema são combinadas através dos operadores genéticos específicos inspirados na Seleção Natural de Charles Robert DARWIN (1809-1882), tais como o Cruzamento e a Mutação. O AG tem como principal característica a geração de um conjunto de soluções ao invés de uma única solução.

Basicamente, esses operadores modificam a população dos cromossomos, e sobrevivem para a próxima geração, os mais aptos dentre essa população de indivíduos. Há uma troca aleatória, porém estruturada, de informações entre esses cromossomos (cruzamento), de modo que, a cada geração, um novo conjunto de cromossomos é criado baseado nos mais aptos da geração anterior. As mutações nos genes dos cromossomos são usualmente geradas de forma aleatória, mas os métodos são determinísticos e em geral podem ser considerados como sendo do tipo genético com regras de mutações determinísticas.

O AG possui 5 componentes básicos: 1) Um método para codificação dos cromossomos, onde os numerais (bits) representam genes); 2) Uma função objetivo "*fitness*" (menor comprimento); 3) Uma população de cromossomos inicial; 4) Um conjunto de operadores para formarem a evolução entre dois cromossomos de populações consecutivas (mutação e cruzamento), que são usados entre as posições das espécies; 5) Parâmetros de trabalho (por exemplo, 2% de probabilidade de mudança no cromossomo) (Grefenstette 1986). A Figura 2 mostra o fluxograma do AG.

Figura 2. Fluxograma do AG.

4. Cálculo do Comprimento do Dedilhado

O AG calcula o comprimento do dedilhado para atribuir um valor de referência à função *fitness* ou função objetivo do AG. A função *fitness* será utilizada pelo SI para encontrar o dedilhado ótimo

O teclado de um piano possui 52 teclas brancas e 36 teclas pretas. A cada tecla branca foi atribuído um número que corresponde à nota e posição no teclado, recebendo valores compreendidos entre –23 a +28 com o dó central sendo igual a zero. Dessa forma podemos obter qualquer relação intervalar entre as notas.

Seja C_{12}, o comprimento do dedilhado para as notas n_1 e n_2, C_{23} o comprimento do dedilhado para as notas n_2 e n_3, e seja $C_{(n-1)(n)}$ o comprimento do dedilhado para as (n-1)-ésimas e n-ésimas notas. Assim, para se calcular o comprimento do dedilhado, representado por *C*, de uma frase musical com *n* notas pode-se utilizar a equação 1.

$$C = C_{12} + C_{23} + C_{34} + \ldots + C_{(n-1)(n)} \qquad (1)$$

O cálculo do comprimento $C_{(n-1)n}$, depende da seqüência das notas e dos dedos. Assim, são quatro as possibilidades das seqüências: 1)movimentos diretamente relacionados, 2)direções de movimentos inversos, 3)dedilhado igual e outra direção das notas e 4)notas iguais e outra direção de dedilhado.

1) Se a nota for ascendente ou descendente em relação a sua antecessora e o dedilhado da nota também o for, ter-se-á uma seqüência de notas e dedilhados numa mesma direção (ascendente ou descendente). Assim, o comprimento deste intervalo é definido como o mostrado na equação 2.

$$C_{12} = |n2 - n1| + |(d_2 - d_1) - 1| + limita \qquad (2)$$

Onde,

n1 = número identificador da primeira nota da melodia.
n2 = número identificador da segunda nota da melodia.
d1 = número identificador do primeiro dedo a ser aplicado.
d2 = número identificador do segundo dedo a ser aplicado.

Acrescentou-se a variável *limita* na equação 2 para penalizar o comprimento do dedilhado. Foram considerados alguns casos para a existência da variável *limita*. No primeiro caso, os dois primeiros dedos são o anular e o indicador e se a diferença máxima entre as notas for maior que 4. No segundo caso, os dedos são o indicador e o médio ou o médio e o anular e a diferença máxima entre as notas for 2. E no terceiro caso, os dedos são o anular e o mínimo e a diferença máxima entre as notas for maior do que 3.

2) Se as notas forem ascendentes e a seqüência dos dedos decrescentes ou então, se as notas forem descendentes e a seqüência dos dedos crescentes então: C_{12} é definido como mostrado na equação. 3.

$$C_{12} = 2 * |(n2 - n1) + 1| + |(d2 - d1) - 1| + |limita| \qquad (3)$$

3) No caso de houver dedilhado iguais, i.e., o mesmo dedo teclando as 2 notas, e com qualquer sentido de movimento das notas, o cálculo é feito usando a fórmula apresentada na equação 4.

$$C_{12} = 2 * |n2 - n1| \qquad (4)$$

4) No caso de ocorrer notas iguais e com qualquer sentido de dedilhado, o peso usado para o cálculo do comprimento do dedilhado é menor, como mostrado na equação 5.

$$C12 = (d2 - d1) \qquad (5)$$

A Figura 3 mostra o esquema geral do Sistema Inteligente (SI) desenvolvido para a ensino do dedilhado pianístico. O usuário têm duas opções: execução pela placa de som ou robótica. O SI possui como blocos principais um Sistema Especialista e um Algoritmo Genético utilizados para a escolha da seqüência do dedilhado. Inicialmente, o pianista deve fornecer ao SI a seqüência melódica. A seguir, o SI verifica se é ou não uma regra existente no SE. Caso afirmativo o SI aciona o SE que retorna o dedilhado da regra. Caso negativo o SI aciona o AG que analisa e prepara a melhor seqüência do dedilhado para a melodia em questão, e as envia para a execução melódica. Neste trabalho não serão mostradas as regras utilizadas no SE.

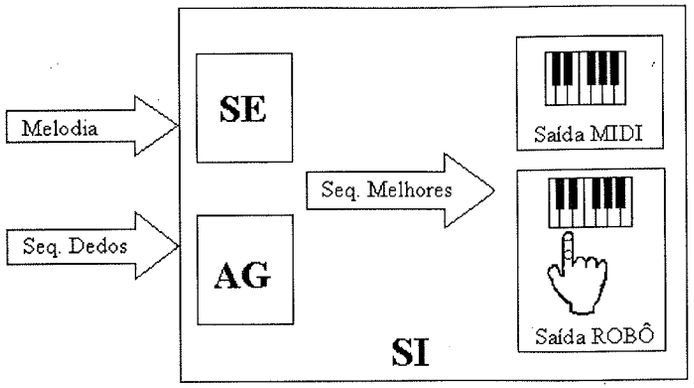


Figura 3. Visão geral do SI para a escolha dos melhores dedos.

5. Escolha do Dedilhado Ótimo com o AG

Utilizou-se uma variação do AG padrão. No AG do SI foram utilizadas duas populações de cromossomos: 1) uma população de 50 cromossomos, denominada População Principal – PP, em que os cromossomos podem se reproduzir e 2) uma população de 10 cromossomos escolhidos, ou População dos Escolhidos – PE, onde são armazenados os 10 melhores cromossomos. Só existe um operador genético no AG que opera diretamente sobre os cromossomos da PE. Basicamente, a cada geração, escolhe-se aleatoriamente um cromossomo da PE, a seguir, escolhem-se aleatoriamente dois genes do cromossomo. A operação genética faz-se mudando a posição entre os dois genes escolhidos. O cromossomo resultante é colocado aleatoriamente na PP.

6. Resultados Obtidos

O SI foi implementado no C++ Builder da Borland®. Testou-se o SI para a execução de algumas melodias simples para piano. Para a melodia de 5 notas mostrada na Figura 4 inicialmente foi gerada a PP (não mostrada neste trabalho). A seguir, foi feito a escolha dos 10 melhores cromossomos da PP para gerar a PE inicial. A PE, mostrada do lado direito da melodia, apresenta os dedilhados em ordem crescente do comprimento da melodia (função *fitness*). As notas foram transformadas em TECLAS 0 4 8 5 6, etc. O número de gerações foi escolhido heuristicamente baseado no número de notas da seqüência melódica. CROM – Cromossomo; COMP – Comprimento; DEDILHADO onde 1->2 significa o polegar na 1ª tecla, o indicador na 2ª, etc.

O número à esquerda de cada coluna representa o número do cromossomo da PP, o valor intermediário refere-se ao comprimento encontrado para a seqüência do dedilhado ao seu lado direito. O cromossomo número 8 da PE é o ótimo (menor comprimento). O

número de gerações foi escolhido heuristicamente baseado no número de notas da seqüência melódica.

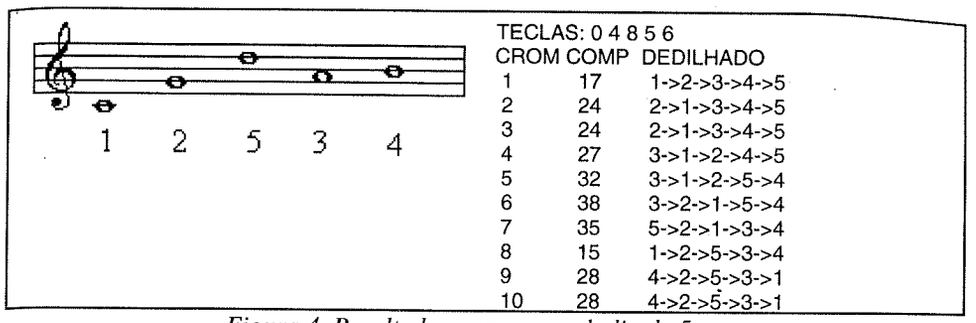


Figura 4. Resultado para uma melodia de 5 notas.

Para uma melodia de 6 notas, pode ser necessário que o AG gere até 720 populações até que se encontre o melhor dedilhado. Neste caso, o AG implementado usou a geração dos cinco dedos iniciais da mesma forma que no caso para uma melodia de cinco notas. À nota seguinte foi gerado mais um dedo aleatório para completar o dedilhado, já que temos seis notas e apenas 5 dedos. A Figura 5 mostra o resultado da escolha dos dedilhados para a melodia. Observe que os melhores resultados só começaram a aparecer a partir da iteração de número 88 e com o Comprimento Médio (CM) de 20,80. O CM é a soma dos comprimentos dividido pelo número de cromossomos. O gráfico abaixo da melodia representa a curva do melhor dedilhado (menor comprimento) *versus* iteração. Quanto mais gerações se passam, melhores são os resultados obtidos.

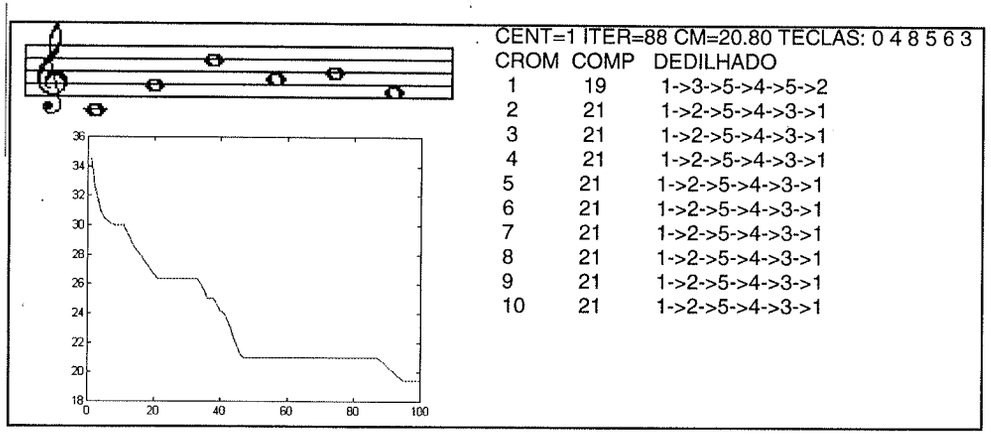


Figura 5. Resultado para uma melodia de 6 notas.

Para um exemplo de 7 notas, obteve-se os resultados apresentados na

Figura 6. Por coincidência um dedilhado ótimo já apareceu na primeira iteração, já que com 7 notas, o AG poderia levar no máximo até 5040 gerações de cromossomos para se chegar ao melhor.

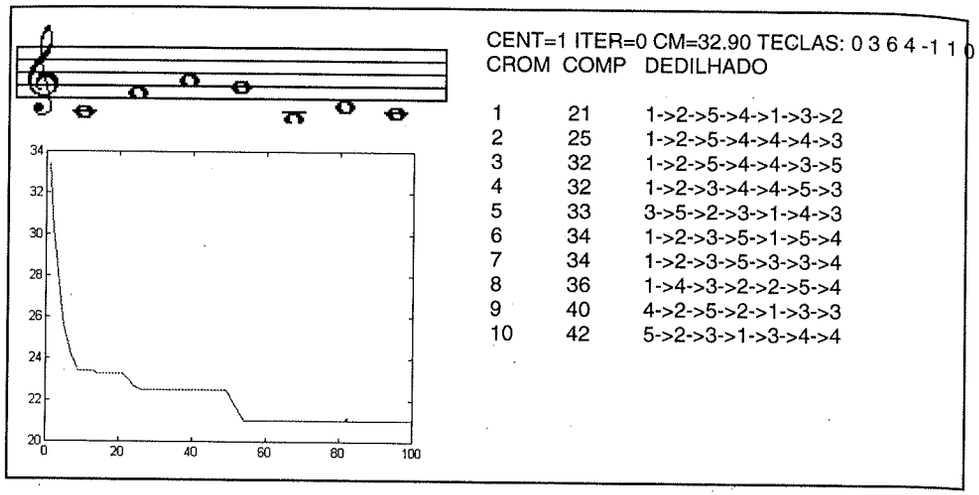


Figura 6. Resultado para uma melodia de 7 notas.

A Figura 7 mostra o resultado para uma melodia com 8 notas. Apenas na iteração 680 é que se chegou ao dedilhado ótimo.

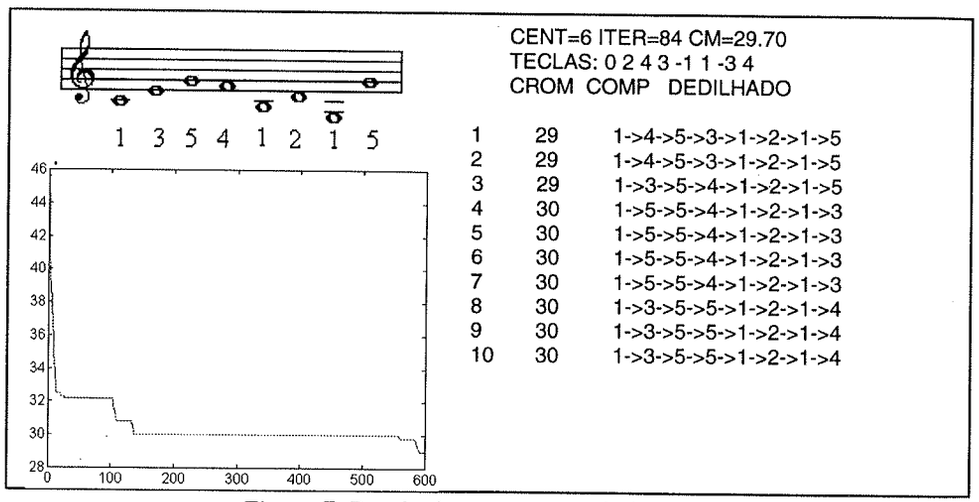


Figura 7. Resultado para uma melodia de 8 notas.

7. Conclusões

Apresentou-se um Sistema Inteligente para ajudar os alunos de piano a fazer o estudo e análise dos dedilhados de melodias para piano. O SI é capaz de mostrar o melhor dedilhado e executar o trecho melódico, apresentando na tela o exemplo prático de como o aluno deve tocar. Espera-se com isso que haja um aumento do ânimo do aluno ao estudo uma vez que o dedilhado complicado e confuso só iria prejudicar o seu desenvolvimento e fazer com que haja uma perda de interesse na prática do instrumento.

Já está em pleno funcionamento um braço mecânico (manipulador robótico) que executa num sintetizador (teclado musical) o trecho melódico, facilitando a visualização do movimento da mão e mostrando todas as passagens de dedos necessárias nos locais indicados pelo algoritmo, o que torna mais prático e objetivo a prática desse instrumento que até então se coloca na posição de um dos mais complexos de todos os instrumentos musicais.

8. Referências

Kaplan, J. A. (1987). Teoria da Aprendizagem Pianística. Porto Alegre - RS. Ed Movimento, 1987.

Encarta (1997). Encyclopedia. Microsoft.

Sayegh, S. I. (1989). Fingering for String Instruments With The Optimum Path Paradigm. Computer Music Journal, Vol. 13, No. 3, MIT.

Holland, J. H. (1992). Adaptation in Natural and Artificial Systems. MIT Press/Bradford Books edition.

Grefenstette, J. J. (1986). Optimization of Control Parameters for Genetic Algorithms. IEEE Transactions Systems, *Man, and Cybernetics*, vol. SMC-16, no.1, pp. 122-128.

DESENVOLVIMENTO DE SOFTWARE EDUCACIONAL PARA A MÚSICA

STR - Sistema de Treinamento Rítmico

Prof. Eloi Fernando Fritsch (UFRGS/ULBRA)
Prof ª Dr ª Rosa Maria Viccari (UFRGS)
Prof ª. Dr ª Zeny Oliveira de Moraes (ULBRA)

Tiago Rubin – (UFRGS/FAPERGS)
Roges Grandi – (UFRGS/FAPERGS)
Luciano Flores – (UFRGS/CNPq)
Gregor Brunelli – (ULBRA)
Willy Schneider – (ULBRA)
Eduardo Campos – (ULBRA)

Universidade Federal do Rio Grande do Sul (UFRGS)
Av. Bento Gonçalves, 9500, Bloco IV Cx.Postal 15064 CEP 91501-970, Porto Alegre, RS
Universidade Luterana do Brasil (ULBRA)
Rua Miguel Tostes, 101, Bairro São Luís, CEP 92420-280, Canoas, RS

Abstract

This project presents STR (Rhythmic Training System), a non traditional implementation of tool for teaching musical theory. It was developed with special care for the aspects of the graphical interface, operation, reproduction of rhythmic patterns, pedagogy, sound resources and other functional characteristics. STR is based on previous systems developed by (fritsch 95,96) - the STI and the SETMUS. The research was developed for Windows 95 hands on Asymetrix'sToolbook II Instructor 5.0. This system can be used for music teachers as reinforcement of theory classes in the following aspects: to help on the musical learning and to wake up the students' interest since STR gathers the functions of musical instrument, score and theory.

The software have nowadays two main modules: the Tests Module and the Rhythmic Calculator Module. In the Tests Module the student interage with the program trying to get the right rhythm the machine performed. The Rhythmic Calculator is a database that stores rhythmic cells of famous music in different styles the student can request its reproduction.

1. Software para Instrução Musical

Na última década, com o aparecimento dos sistemas multimídia, vários programas de computador para o ensino musical foram idealizados e desenvolvidos com a finalidade de tornar a teoria e a percepção musical acessíveis aos usuários que desejassem estudar música através de computadores. Com a utilização das tecnologias de hipertexto, programação visual orientada a objetos e a multimídia, estes programas, que antes rodavam em telas verdes e circuitos que geravam o som monofonicamente , começaram, gradualmente, a mudar para telas gráficas coloridas e sons de alta fidelidade, despertando o interesse de professores e estudantes de música. Devido a grande capacidade de memória, o poder de processamento e a criatividade dos projetistas de software, foi possível a criação de vários sistemas para o ensino alternativo da teoria e percepção musical. A descrição de sistemas de instrução musical como o Auralia, Claire, Euterpe, Listen, MusicLab, Music Lessons e Music Mentor podem ser encontrados em (Ratton, 1997).

2. STR - Sistema de Treinamento Rítmico

O projeto do STR provém da idéia inicial de criar um grande sistema de computador que englobe a teoria musical em suas diversas áreas testando cada módulo do sistema em escolas de música em processo de informatização. Até então, foram desenvolvidos e aprimorados os sistemas SETMUS - Sistema Especialista para Teoria Musical (originalmente desenvolvido para Macintosh e depois reprogramado para Windows) e STI - Sistema de Treinamento de Intervalos (originalmente desenvolvido em HyperCard e posteriormente atualizado com rotinas de comunicação MIDI através do HyperMIDI) (Fritsch, 1995, 1996).

O STI foi implantado na escola Prelúdio da UFRGS em 1997 onde foi utilizado nas aulas de teoria musical possibilitando que os alunos realizassem vários testes com o sistema no reconhecimento e percepção de intervalos melódicos. Após o sucesso dos teste realizados com o STI na escola de música Prelúdio, ficou clara a necessidade da criação de outros módulos para o ensino de teoria musical utilizando a plataforma PC. O Macintosh praticamente não é utilizado em escolas de música no Brasil. Por essa razão o SETMUS foi portado para o ambiente Windows e iniciou-se o projeto de um novo sistema em PC para treinamento de rítmos.

3. O Macintosh como plataforma para o STR

De acordo com os resultados obtidos nos projetos anteriores e na pesquisa realizada concluímos que o Macintosh é um ambiente adequado para o desenvolvimento do STR (Fritsch, 1996).

Do ponto de vista ténico, o HyperMIDI possibilita uma programação rápida e eficiente do protocolo MIDI sendo uma ferramenta poderosa no tratamento dos eventos MIDI.

Para informações sobre a programação HyperMIDI pode-se consultar o HyperMIDI manual version 2.0 escrito por (Redmon, 1990).

Apesar de iniciarmos os primeiros testes e prototipações para averiguar a possibilidade de implementar a ferramenta STR no Macintosh através do HyperCard, optamos em programar o STR na plataforma PC, pois o produto final atenderia um número maior de usuários. O protótipo no Macintosh serviu para confirmar de que era possível implementar o sistema no Hypercard e rotinas MIDI através da pilha HyperMIDI.

O ToolBook foi a ferramenta escolhida para a implementação do STR por ser um software de autoria com grande flexibilidade na programação gráfica e sonora.

4. Emissão de notas musicais utilizando o ToolBook

Primeiramente, foram utilizadas rotinas que usavam amostras digitais dos instrumentos para tocar música. Dessa forma foi possível prototipar alguns módulos do sistema STR, como o módulo de ditado rítmico, devido a rápida programação e utilização dos arquivos Wave. Assim foi possível testar a funcionabilidade de algumas rotinas, inclusive a execução sonora.

Os comandos de sons que estão sendo utilizados no STR são o PlaySound e o MmPlay. O comando PlaySound toca um arquivo em forma de ondas sonoras.

O comando PlaySound funciona de forma melhor para arquivos com tamanho menor que 100Kb. Esta função necessita do acesso a biblioteca *MMSYSTEM.DLL* e a um driver de som. *MMSYSTEM.DLL* é um arquivo standard DLL no Windows.

O comando MmPlay executa um Clip. Clip é um evento de som ou imagem que é importado para dentro do ToolBook II pelo gerenciador de recursos. Se o Clip não estiver aberto, o ToolBook II automaticamente abre o clip antes de tocá-lo e fecha quando a operação termina. A utilização do tempo de cada nota é feito mediante a utilização do comando mmPlay.

Até então trabalhou-se apenas com arquivos Wave. No entanto, como a pesquisa está em desenvolvimento, já estão sendo construídas DLLs para o gerenciamento de eventos MIDI. Com a utilização da MIDI, teremos uma economia de memória e uma temporização ainda mais exata das figuras rítmicas de menor valor. (Selfriedge-Field, 1997), (Yavelow, 1992), (Young,1996).

5. O STR para Windows

O STR inicia com um menu onde é possível selecionar qual será o módulo do sistema a ser utilizado. Os módulos em desenvolvimento são constituídos por exercícios que o aluno realiza para desenvolver suas potencialidades rítmicas. Além de exercícios, o aluno também pode acionar botões que os levam a telas e procedimentos que explicam o conteúdo. Serão abordados três módulos em desenvolvimento: teoria dos ritmos, o ditado rímico e a calculadora rítmica.

O **módulo de teoria do ritmo** apresenta o aluno à grafia universal da literatura musical. Foram desenvolvidas telas, pela professora de educação musical Dr ª Zeny Oliveria

de Moraes, com a finalidade de familiarizar o aluno com as figuras musicais e suas relações. As figuras de 1 à 3 são exemplos de como as figuras rítmicas foram didaticamente projetadas no sistema para um melhor entendimento.

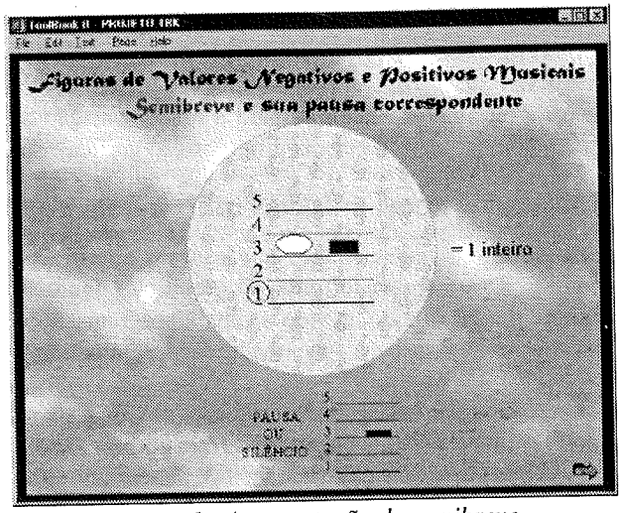


Figura 1 - Apresentação da semibreve.

O **módulo de ditado rítmico** pode ser visualizado na figura 4 apresentando uma interface gráfica com um menu superior de opções, uma pauta musical, um menu de instrumentos, controle de volume e de tempo.

Para explicar o funcionamento do sistema iremos detalhar as funções acionadas pelos principais botões do STR.

O botão *Iniciar Teste* inicia o módulo de testes de performance rítmica do aluno de música. Ao pressioná-lo, o STR produz uma seqüência MIDI ou Wave de acordo com o módulo de configurações (figura 6) sempre utilizando a nota dó4, em compasso quaternário 4/4. Ao mesmo tempo, abre uma *palette* de figuras de tempo e de figuras de pausa, da semibreve à semifusa, e da pausa da semibreve à pausa da semifusa, respectivamente, conforme a Figura 5. Como pode ser observado na Figura 5, as figuras pontuadas também foram incluídas no *pallete* de símbolos musicais.

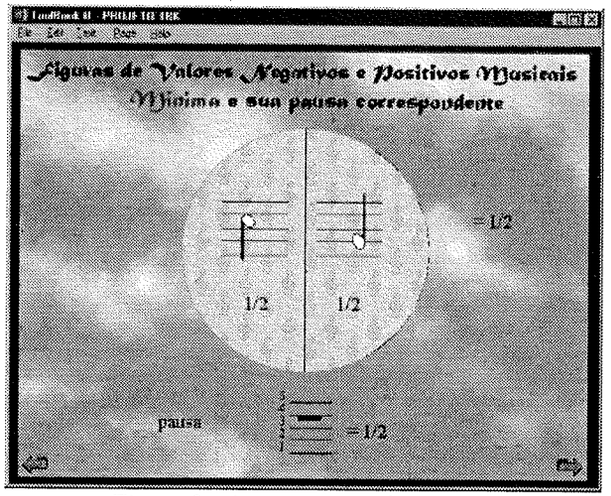


Figura 2 - Apresentação da mínima.

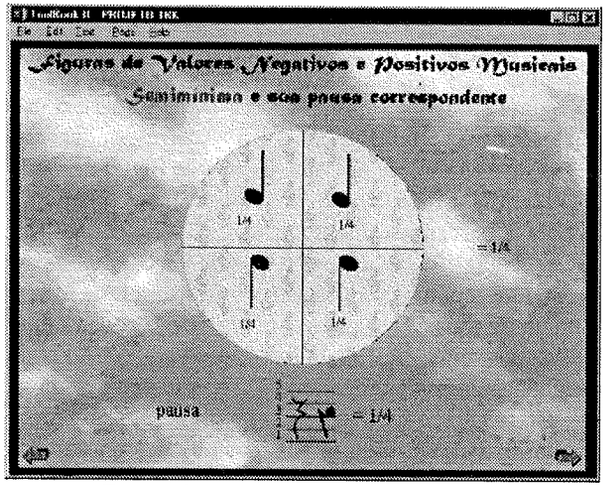


Figura 3 - Apresentação da semínima.

O aluno, então, deve pressionar os botões das figuras que correspondem à seqüência que escutou. Ao acertar uma figura, esta aparece na partitura. Ao errar, poderá repetir a tentativa escolhendo outra figura. Após o usuário selecionar todas as figuras que ouviu, o teste estará concluído pois o compasso estará completo pelas figuras corretas. A caixa de diálogo, *Teste concluído*, aparecerá sempre que o compasso for corretamente preenchido pelas escolhas do estudante de música. O usuário poderá realizar um novo teste selecionando no botão Iniciar Teste.

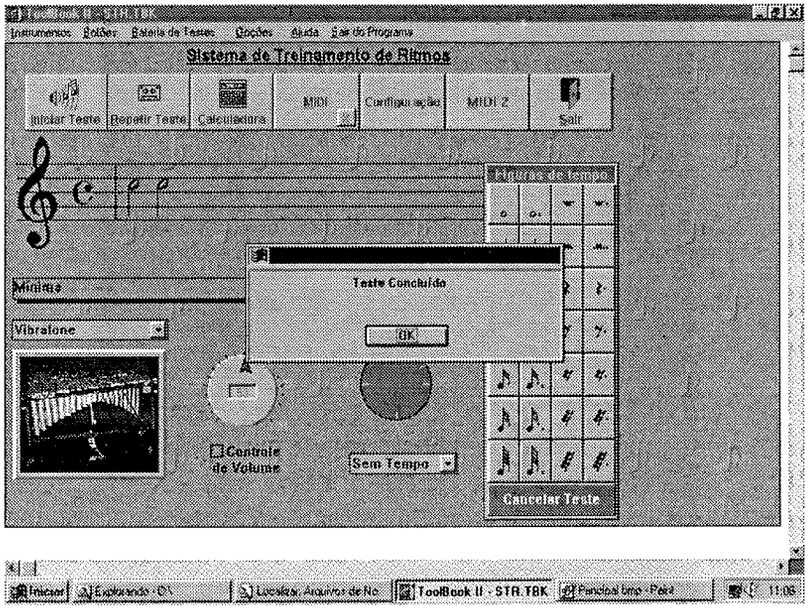


Figura 4 - Tela do STR, após a execução de um ditado rítmico

O *pallete* de figuras de tempo e pausas foi elaborado para o usuário ter uma maior interatividade com o sistema. O STR produzirá o som correspondente ao valor da figura de tempo e tentará selecionar o símbolo gráfico no pallete de figuras. O botão *Repetir Teste* repete a seqüência MIDI tocada no teste, caso o aluno precise escutá-la novamente. Isto é fundamental para a percepção dos iniciantes. É natural que a pessoa precise ouvir mais de uma vez a seqüência rítmica antes de selecionar os símbolos correspondentes no *pallete*. O estudante poderá repetir o teste quantas vezes desejar a critério de seu professor. Isto permite que o aluno de música eduque o ouvido sem o constrangimento da pressão de uma sala de aula repleta de alunos realizando um ditado rítmico. Ao contrário do aluno ficar inibido, ou então acuado pelo medo de errar em sala de aula, no computador ele tentará acertar sem pressões externas.

A **configuração do sistema** altera os parâmetros dos testes como a quantidade das notas, e o tempo. Este módulo de configuração pode ser utilizado pelo professor de música para selecionar várias opções que irão alterar o ditado rítmico realizado no STR. Conforme a figura 6, pode-se observar que o usuário tem a possibilidade de alterar o nível de dificuldade dos testes realizados pelo STR. Portanto, é possível escolher quais as figuras que mais irão se repetir nos testes.

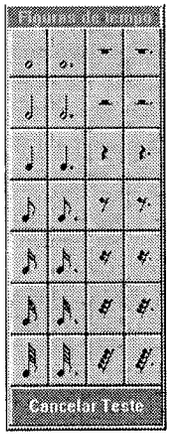


Figura 5 – Conjunto de símbolos musicais possíveis de serem selecionadas pelo usuário durante o teste.

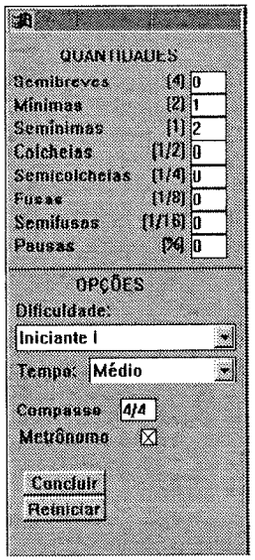


Figura 6 - Caixa de diálogo para seleção de níveis de dificuldade no ditado rítmico.

O nível de dificuldade pode ser alterado a critério do professor. Por exemplo, um aluno que estiver iniciando na percepção rítmica poderá utilizar o nível Iniciante que trará figuras com uma duração maior. Poderá ser selecionado o tempo utilizado para o usuário tocar as notas do ditado. Também foi implementada opção para a escolha do compasso a ser utilizado no ditado rítmico, conforme figura 6. A opção Metrônomo, liga ou desliga o

metrônomo durante o ditado do STR. O "x" significa que o metrônomo está ligado. Para concluir a seleção de parâmetros basta clicar na opção concluir.

A calculadora rítmica é uma tentativa de desenvolver um sistema de computação que auxilie o aluno na educação de seu ouvido musical. A calculadora representa um outro módulo do sistema onde o aluno poderá selecionar parâmetros como o estilo, o andamento e o compasso. De acordo com essa seleção, será executada uma música em MIDI que tenha todas estas características.

Todas as melodias foram transpostas para a tonalidade de Dó e podem ser ouvidas pressionando-se o botão Ritmo (vide figura 7).

O objetivo do botão Melodia é dar ao aluno um exemplo conhecido de músicas de um determinado estilo, que o ajude na memorização rítmica do estilo pela associação com o trecho musical reproduzido.

O botão ritmo tem a função de executar o ritmo utilizado na melodia apenas numa nota, ou seja, serão executadas as figuras de tempo utilizando sempre a nota Dó4. Dessa forma, o aluno poderá dar uma maior atenção ao ritmo sem se preocupar tanto com a melodia. Isso poderá facilitar a percepção de detalhes rítmicos, principalmente pelos iniciantes.

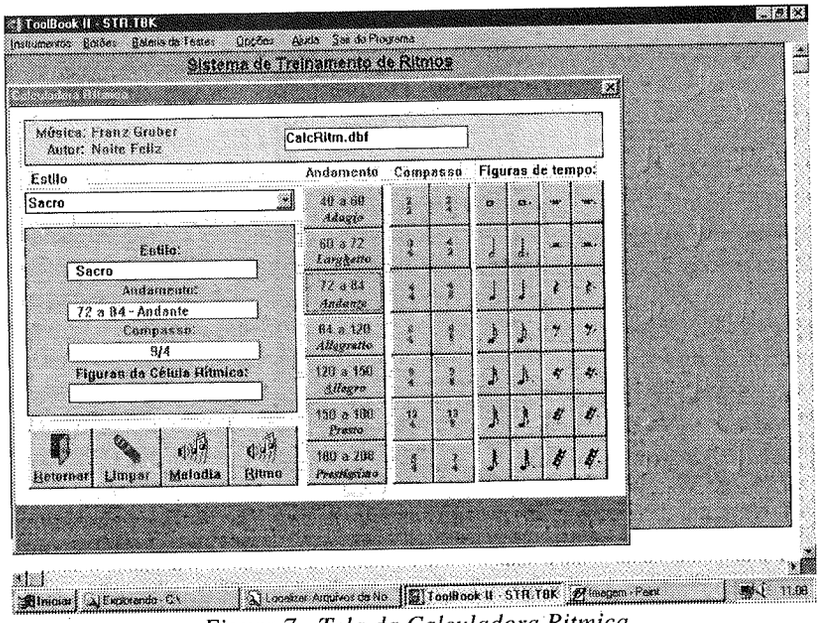


Figura 7 - Tela da Calculadora Ritmica

Para melhor guardar e recuperar os dados referente à calculadora musical foi implementado um banco de dados. Anteriormente, eram gravados os dados de gerência e controle dos exemplos musicais MIDI em script. Esta transformação, de

script para tabelas de banco de dados, se faz necessária, uma vez que as tabelas permitem receber receber novos ritmos. Além destas vantagens, a indexação oferecida por tabelas permite versatilidade. Foi escolhida a estrutura dbf padrão dBase por estar integrada aos métodos do ToolBook II da Asymetrix.

6. Conclusão

O desenvolvimento de software educacional para a música visa pesquisar e construir novos sistemas que auxiliem no aprendizado musical. Para isso é necessário que os ambientes utilizados na implementação disponham de rotinas musicais eficientes e de interfaces interativas.

O STR foi concebido para estudantes que já possuam uma base musical e desejam exercitar seus conhecimentos em ritmo a fim de aprimorar sua percepção. O sistema foi desenvolvido para servir de apoio às aulas de teoria musical que envolvam ritmo.

Os resultados finais desta pesquisa provaram que a utilização de software de autoria para a construção dos sistemas é uma alternativa viável na interação do usuário com conceitos musicais sobre ritmo. O software de autoria também mostrou ser muito eficiente na realização de módulos onde os conhecimentos possam ser exercitados pelo músico aprendiz. A flexibilidade na implementação de várias rotinas gráficas como as notas musicais e o pentagrama torna o Tool Book, em particular, uma ferramenta indicada para o desenvolvimento de aplicações musicais voltadas à área de ensino.

O STR está implementado para arquivos de áudio digital. A programação do sistema avança para a incorporação de funções MIDI com o objetivo de economizar memória e no futuro utilizar o sistema via internet.

Está previsto o desenvolvidos de outros módulos de exercícios rítmicos para o STR, bem como a união deste sistema com os demais programas musicais já em funcionamento que são fruto das pesquisas anteriores. Também está programado um teste piloto na escola de música do projeto prelúdio da UFRGS para a validação do sistema em sala de aula e como sistema de apoio para estudo individual. De posse desses resultados, os sistema será aperfeiçoado e implantado nesta mesma escola.

Agradecimentos

Fundação de Amparo à Pesquisa do Rio Grande do Sul - FAPERGS
Universidade Luterana do Brasil - ULBRA
Universidade Federal do Rio Grande do Sul - UFRGS
Conselho Nacional de Ensino e Pesquisa - CNPq

Bibliografia

CAMP, Victoria. **Making Music on Your PC**. Grand Rapids: Abacus, 1997. 348p. Il.

CARY, Tristam. **Dictionary of Musical Technology**. New York: Greewood, c1992. 542p. Il.

FRITSCH, Eloi F. & VICCARI, Rosa M.1995. **SETMUS** - Sistema Especialista para Teoria Musical. II Simpósio Brasileiro de Computação e Música. Canela - RS.

FRITSCH, Eloi F. 1996. **STI** - Sistema de Treinamento de Intervalos. III Simpósio Brasileiro de Computação e Música. Recife - PE.

FRITSCH, Eloi F. 1995. **Música Computacional**: A Construção de Sistemas de Computação para Música. Logos. Canoas, v.7, n.2 .Out/95.p.76-88.

HELMSTETTER, Anthony. **Web Developer's Guide to Sound & Music**. Scottsdale: Coriolis Group Books, c1996. 317p. Il.

HEYWOOD, Brian. **PC Music Handbook: windows 95 compatible**. 2.ed. Kent: PC, 1996. 216p. Il.

MESSICK, Paul - **Music Applications in C++ - MAXIMUM MIDI**, Editora Manning, Greenwich - EUA, 1995.

PENFOLD, R.A. **Advanced MIDI User's Guide**. 2.ed. Kent: PC, 1995. 184p. Il.

REDMON, Nigel. **HyperMIDI version 2.0**. Torrance, Califórnia, 1990.

RATTON, Miguel. **Computador & Música**. Informus, Rio de Janeiro, CD-ROM, 1997.

SELFRIEDGE - FIELD, Eleanor. **Beyond MIDI: the handbook of musical codes.** Cambridge: MIT, c1997. 630p. Il.

YAVELOW, Christopher, **Music & Sound Biblle**, San Mateo, California: IDG Books WordWide, Inc., 1992.

YOUNG, Robert. **The MIDI Files**. London: Prentice Hall, 1996. 318p. Il.

composições

Comme il batacchio della settima campana (1997)

(for tape)

Riccardo Artico
University of Roma "La Sapienza"

The will to extrapolate from a piano a new spectral and spatial dimension drove me to the composition of " Come il batacchio della settima campana" [Like the clapper of the seventh bell].
It seemed to me very stimulating the idea of putting myself in front of an instrument so important for our musical tradition and using it to create, by computer ways, a new acoustic space.

First step was to sample sounds from a piano, but I was interested in use of the tailpiece without using keyboard and hammers as a striker.

I used a little bell without clapper to produce a series of samples, obtained scratching and striking chords or inside walls of harmonic box of a grand-piano.

The consequent transformation of these samples is obtained through the Csound program by filtering, transposing, time stretching, overlapping, reverberating, spatializing, etc. The final mixing is realised with the audio-board software Triple-dat.

I followed formal rule: 1) Source bondings: sound materials were elaborated to understand, o misunderstand, a relationship with the primal source; 2) Spatialization: allocation between channels and reverberation aim to create a variable and multiform acoustic space; Gesture: sound and its time development responds at movement and physical gestual expectations that are natural or artificial connected.

Maybe a trip is the metaphor that represents the final result of this work; a trip inside the piano, intended as a place and physic space to explore and listen. To listen, which is just like filling an empty space. Who listens should recreate that hammers mechanism absent, absent 'like the clapper of the seventh bell'.

Poema Negro (1997)

(para clarinete em Sib e CD)

Gilberto Carvalho
Escola de Música
Universidade Federal de Minas Gerais
agmc@dedalus.lcc.ufmg.br

A obra "Poema Negro" é baseada no trabalho homônimo do poeta brasileiro Augusto dos Anjos (1988-1913). Neste, um personagem, após reflexão de caráter introspectivo, é assaltado por um conjunto de visões "negras". Ao término destas, reconquistando a lucidez, retoma os pensamentos iniciais, desta vez modulados pelas novas experiências.

Do ponto de vista da construção formal a obra segue as articulações sintáticas e semânticas do poema. Isto significa que nela é ensaiada uma integração entre o material musical (em seus diversos parâmetros) e o material poético, integração que possibilita uma projeção formal do poema na obra musical.

Com relação à escritura instrumental, esta utiliza quase a totalidade da tessitura do instrumento e grande parte do seu potencial de dinâmicas, possibilidades microtonais e transições timbrísticas entre alturas e ruídos próprios à estrutura acústica do clarinete.

No que diz respeito à parte eletrônica, foi elaborada através de sons puramente sintéticos e de sons pré-gravados de clarinete. Dentre os programas e sistemas utilizados para sua elaboração constam o Csound (síntese e processamento de som), o ProbSys (composição) e o SICX (composição e conversão de formatos). Estes dois últimos sistemas foram implementados pelo autor na Escola de Música da UFMG.

Points of No Return (1997)

(for two-channel tape)

Chin-Chin Chen
University of Illinois at Urbana-Champaign
c-chen8@uiuc.edu

Points of No Return shifts between two different environments or landscapes. To achieve this, sounds were divided into two categories according to their nature and timbre; but as the piece goes on, some sounds from one environment also occur in the other one. Points of No Return employs music concrete techniques and digital editing and processing. Only at a very late stage is electronically generated sound incorporated to color some dramatic moments. Points of No Return is divided into 5 sections, alternating between two different landscapes.

Multiple Reeds (1994)

(for saxophones and tape)

Rodrigo Cicchelli
Escola de Música
Universidade Federal do Rio de Janeiro
Rodcv@openlink.com.br

"Voyage to the Interior of the Saxophone" could well be a sub-title for this piece. But before we start considering the work itself, we must first ask: what is the saxophone?

A metallic conical tube, with holes topped by leather pads, propelled by human breath, making a single bamboo reed vibrate. It could also be seen as a cross between an oboe and a flute, capable of producing an extraordinary variety of timbres, from very quiet and delicate sounds to extremely powerful and penetrating multiphonics. The flexibility of sound producing techniques on this instrument makes it of unique interest, for we can isolate quite convincingly different actions primarily thought to work in conjunction. For instance, we may want to explore breath sounds through the tube, filtered by ordinary fingerings, but without generating pitches; we can, however, extend the pitch producing techniques by arranging special throat and finger positions in order to produce richly irregular harmonic sounds; or we may want to explore the more percussive side of the instrument by exaggerating key clicks.

This experimental approach is tackled in conjunction with the electronic part, where not only did I use saxophone sounds (transformed or not), but also other reed instruments. Further sounds on tape were chosen as extensions of the saxophone materials, for instance: metallic tube --> cymbals resonance, breath sounds --> filtered white noise, key clicks --> drums and other percussive instruments. The computer was used to analyze source sounds and create transformations between these various sources through several filtering processes. It was also used as the environment where all electronic sounds were mixed down and transferred onto a digital tape.

The piece is divided into three movements, exploring different expressive capabilities of the saxophone family. The first (on the soprano) is assertive, the second (on the tenor), a cantabile, and the third (first on the tenor and then on the soprano), a toccata-like game. Echoes of different styles are hinted, but never used as overt quotations. It is as if I wanted to create a voyage through a very personal world. Some elements in this world we share, others, you are invited to discover.

Multiple Reeds is dedicated to the saxophonist Stephen Cottrell, who first performed it in May 1994 at the Sainsbury Centre of the University of East Anglia.

COMBINA C1917 (1997)

Homage to Pietro Grossi
(for tape)

Roberto Doati
Centro di Sonologia Computazionale, University of Padova
Doati@tin.it

1917: Pietro Grossi is borning in Venice.

1972: the RAI (Italian National Broadcasting) broadcasts "C'è musica e musica", a contemporary music series edited by Luciano Berio. In one of these television presentations Pietro Grossi, at that moment at the Biennale di Venezia Music Festival, is invited to speak about his computer music experiences. He is the first to promote such field in Italy.

1977: I am starting to follow the Pietro Grossi courses on Computer Music at the "Istituto di Elaborazione dell'Informazione" of CNUCE-CNR in Pisa. The most important teaching I receive is concerning the fact that the formalization of a compositional process is not limiting at all the creativeness of the composer, but extend it. I cannot forget the great emotion I felt the day I write and see my very first and trivial FORTRAN program running.

1997: I am "rebuilding" some of the commands that were part of the TAUMUS software realized in Pisa by Grossi and use them to transform those 100 seconds of the Berio television broadcasting. The Pietro Grossi and Virgilio Boccardi - the interviewer - voices and the very "naif" sounds of the TAU2 - the minicomputer with DAC used by Grossi - are granularized with a TAUMUS algorithm simulation. It generates simple combinations with 19 elements and 17 sub-elements. So the three different materials are submitted four times (each time with different values as concerns grain duration, starting point of file reading, etc.) to obtain twelve voices. I then time stretched - three times - the broadcasting to get the formal structure which reflects the "counterpoint" between Grossi, Boccardi and the TAU2. All of the sound events are achieved through reading and filtering with random controlled parameters the twelve voices. Three times the Berio's voice in the television broadcasting is superposed to the TAU2. So I used it as amplitude modulation and filtering control signal over the TAU2 "voice".

Thank you to Davide Rocchesso for the combinatorial algorithm and for BaBo.

El reloj del viento (1998)

(for tape)

Silvio Ferraz
Laboratório de Linguagens Sonoras
Comunicação e Semiótica - PUC-SP
sferraz@exatas.pucsp.br

Recentemente me deparei com uma escultura de Juan Miró intitulada "El reloj del viento": uma caixa de madeira com uma colher espetada fundidas em bronze. Imaginei esse dispositivo como um mecanismo que marcaria o tempo de acordo com a velocidade e intensidade do vento batendo na colher; um vai e vem bastante irregular. E esta foi a idéia que acabou guiando esta peça que de certa forma tem a obra de Miró sempre em mente: uma seqüência rítmica irregular sobre a qual diversas transformações sonoras seriam realizadas

Mas Miró não aparece nesta peça apenas na idéia do relógio. Sempre me interessou o modo como realizava seus quadros: deixava diversas telas espalhadas pelo ateliê e de tempo em tempo as manchava limpando os pincéis ou com uma pincelada qualquer, até que um dia uma daquelas telas despontava como um possível quadro. esse processo que Miró cita em diversas de suas entrevistas é que foi usado para compor "El Reloj del Viento".

Como as telas de Miró, uma seqüência de sons de tamtam foi gravada e espalhada em diversas possíveis peças, e a cada dia uma ou outras dessas peças eram escutadas e trabalhadas basicamente por deformações da seqüência original, por acréscimo de outros sons ou ainda pelo cruzamento entre esses sons. Até que um dia diversas delas começaram a convergir resultando naquilo que seria o mapa rítmico da peça...mapa esse que vale dizer que foi totalmente destruído não aparecendo no resultado final da peça.

Três softwares específicos foram usados na composição dessa peça: AudioSculpt e os ambientes de programação por objetos Patchwork e MAX.

Todos os sons utilizados na composição decorrem de transformações realizadas a partir de filtros que foram gerados em programas realizados em Patchwork. Neste caso o software foi utilizado para criar programas que geram arquivos de texto seguindo o protocolo dos filtros, síntese cruzada e outros procedimentos (time stretch, transposições e granulação de sons) habilitados nas versões mais recentes do software de análise, tratamento e síntese AudioSculpt desenvolvido pelo IRCAM.

Fundamentalmente, este uso do Patchwork permitiu criar uma estrutura que seguia padrões da composição instrumental, permitindo pensar em campos harmônicos (estruturas harmônicas) e em estruturas rítmicas, os quais serviram como um mapa para a composição da peça. Vale lembrar que todas essas estruturas decorreram da análise espectral e rítmica dos três principais

arquivos de som utilizados na peça: uma seqüência de sons de tamtam, trechos de uma conferência e uma seqüência de sons decorrentes de um cinzeiro de barro raspado com uma flauta indígena.

Associado a esses dois softwares, foram também realizados dois programas em ambiente MAX: uma espécie de um granulador de sons (disparando arquivos em aiff) e um outro que disparava diversos samples a partir de arquivos MIDI criados com auxílio de Patchwork. Porém, para evitar a característica repetitiva do sons disparados via MIDI, o resultado deste processo também foi filtrado posteriormente em AudioSculpt.

Alma Latina (Soul of a Latin) (1996, rev. 1997)

(for tape)

Rajmil Fischman
Keele University, UK.
r.a.fischman@mus.keele.ac.uk

... this Latin soul hopes that there is still warmth and a reason to dance, that not all is lost and that, after all and in spite of thoughtless acts and dangerous ventures, life can still go on.

RF

Technical Aspects

Alma Latina was composed at Keele University Music studios using a *Composers' Desktop Project (CDP)* computer workstation connected to a *Soundscape* unit. Final equalisation was carried out with *Focusrite* on a *Pro-Tools* system.

Technically, this piece aims to create a seamless transition between recognisable anecdotal sounds from a Latin American aural environment, and abstract sonic objects. In order to achieve this, various processes are used which bridge between the two extremes, such as the total or gradual transformation of anecdotal material using spectral techniques and the creation of rhythmic complexes in which anecdotal and abstract interact.

Specific examples of processes include vocoding (the imposition of the time-varying spectral envelope of one sound onto another), spectral interleave, which consists of interleaving groups of spectral frames belonging to different sounds and spectral tracing, a process which retains the most prominent partials at any given moment. The computer was also used in order to carry out filtering, brassage, delay, transposition, complex spatial movement with doppler shift and standard mixing, cutting and splicing.

Alma Latina was one of the pre-selected finalists in the Quadrivium Programmatic Music category at the 24th International Competition, Bourges.

Delirium tremens (1997)

(para cavaquinho e fita)

Sérgio Freire
Escola de Música
Universidade Federal de Minas Gerais
sfreire@dedalus.lcc.ufmg.br

O ponto de partida desta composição foram idéias musicais compostas em 1992 para cavaquinho solo, e que por si só não foram capazes de motivar a criação de uma obra completa. A célula geradora destas idéias foi a exploração de dissonâncias envolvendo cordas presas e soltas, além do uso do tremolo nas quatro cordas. Estas idéias foram retomadas em 1996, à luz da síntese digital, principalmente do algoritmo de Karplus-Strong.

A dualidade contida nesta célula geradora foi estendida para a nova instrumentação – instrumento x fita magnética -, e para o uso contrastante de materiais sonoros: pequeno instrumento acústico x instrumento virtual de dimensões variáveis, uso de sons da família do violão x uso de sons vocais, sons com diferentes níveis de transformação e deformação.

Forças compulsivas conduzem o discurso musical, baseado em um reduzido número de materiais sonoros, gerando situações musicais caracterizadas por gestos exacerbados, fragmentações, deformações, coexistências insólitas, nas quais a relação entre o instrumento e a fita varia da total estranheza a uma grande similaridade. A parte em fita magnética mantém um caráter instrumental (valorizando mais as alturas do que texturas) em boa parte da obra.. A introdução é livremente baseada na obra Continuum, de G.Ligeti.

O algoritmo de Karplus-Strong foi explorado em diferentes combinações de freqüência fundamental, de tamanho do buffer e do método utilizado para filtragem, gerando deste modo uma família de sons mais variada do que sua simples aplicação permite.

O material sonoro restante vem do processamento de sons pré-gravados: da expressão falada 'delirium tremens' e de algumas amostras de tremolos do cavaquinho. As técnicas utilizadas foram a variação de freqüência derivada da alteração da velocidade de leitura das amostras, phase vocoder e algoritmos de espacialização, alguns com efeito Doppler. O programa utilizado para toda a composição foi csound, e a mixagem foi realizada em gravadores multipista, nos laboratórios de música e tecnologia da Escola de Música da UFMG.

VOCI DALL'ALDIQUA` (1997)

(for tape)

Diego Garro
Keele University, UK
d.garro@mus.keele.ac.uk

The title 'Voci dall'Aldiqua' is a play on words that means approximately Voices from the material world. Along the piece, human voices and non-human sounds chase each other, blend and struggle, depicting a separation between natural and supernatural which no longer exists. Sometimes the 'voice' is a human voice, clearly recognisable, either speaking or singing. Sometimes the human voice is deeply transformed becoming almost unrecognisable. Other times the 'voice' is, metaphorically speaking, the echo of the material world responding to the human call (or curse?). The finale features the triumph of human voice singing an angelic chant which, nevertheless, dissolves in a sort of self-decomposition: does transience deprives human attributes from beauty and spirituality?

The piece was composed between October 1997 and January 1998 at Keele University using the computerised facilities of the Music Department. In composing 'Voci dall'Aldiqua' the role of the computer was decisive both for the sound transformations required and for the final layout of the piece, including the spatialisation and the overall stereophonic design. The general musical stream often suggested modifications on the sound-transformations previously performed, hence the importance of having used flexible computer sound processing tools with which such adjustments and iterated experimentations can be done in a non time-consuming fashion. In using computer programs for audio-processing, I often experimented with time-varying techniques in order to obtain sounds which dynamically change their morphology. The finale of the piece, for example (featuring a fortissimo vocal choir along with its disintegration), was realised using a time-varying 'shuffling' Plug-In. The voice fragmentation that ends the piece is the result of changing the numerical value of various parameters (feedback, fragment duration, randomised pitch) when applying this processing tool to the vocal fade-out. Here follows a short list of software used in composing 'Voci dall'Aldiqua':

Composers' Desktop Project software, running on IBM compatible PC:
. time domain techniques
. frequency domain techniques on phase-vocoder data

Digidesign PRO-TOOLS running on Machintosh Power-PC:

. varios 'Plug-Ins'
. panning and stereophonic design
. final sound-layout, mixdown and mastering
. C-Sound synthesis

corrente... (1998)

(for tape)
(from "Fluss durchs Ohr")

Thomas Gerwin
ZKM, Karlsruhe
tg@zkm.de

This year's large " Landesgartenschau" of the state of Baden-Wuettemberg, Southwest Germany takes place in Plochingen. A new CD "Fluss durchs Ohr" (River/Flow through the ear) was composed by Thomas Gerwin, a soundscape journey along the Neckar river from the spring up to the flowing together with Rhine river. Ten sound postcards are created from typical places on the line, movement 11 (corrente...) on the Cd is freely composed purely from water sounds which are recorded around, above and inside the river and then processed in the computer music studio called inter art project.

Europa (1997)

(for tape)

Graham Hadfield
City University, London
P.Hadfield@city.ac.uk

Europa is one of a group of four pieces whose titles are the names of the largest moons of Jupiter. The composer used, as an impetus for composition, descriptions of the natures of the four satellites *Ganymede*, *Callisto*, *Io*, and *Europa*, named here in order of size from the largest to the smallest. The durations of the pieces reflect the relative proportions of the satellites: *Ganymede* has the longest duration (largest moon) and *Europa* has the shortest (smallest moon). The listener is free to construct a synaesthesic image from the music which might have some supposable correspondences with the moons' descriptions, but the pieces are not intended to be superficial imaginary soundscapes of the moon's environments. The composer allows his repertory of sonic material to include more abstract, and remote surrogate sounds (sounds surrogated from their acoustic causal source). In describing briefly the nature of *Europa* (the satellite) below, the listener is informed of a small, although significant, part of the poiesis of the piece. Although it is impossible for the composer directly to communicate the description through the acousmatic medium (without recourse to human-uttered language) it is likely that vestiges of the composer's poietic image induced by the description will have resonances in the listener's esthesis. One might say that a correlation, albeit intangible, could occur between the poietic image and the esthesic image; the composer's image and the listener's image respectively. Whilst it is not possible for the listener to experience the composer's image in esthesis, by detailing what is, in part, the cause of the poietic image, the listener is given freedom to construct an esthesic image which, itself in part, has the same cause as that of the poietic. *Europa* has a very bright, frozen crust. In some parts it consists of small hills, and in others it consists of long cracks, some of which are curved and others which are straight. It is conjectured that there might be a liquid, *warm ice* ocean beneath the thin water ice crust, and there might also exist water geysers.

Variasjonar over ei stille (1997)

(Silence and Variations)

(for tape)

Risto Holopainen
risto.holopainen@notam.uio.no

Variasjonar over ei stille (silence and variations) is based upon two poems by the composer, originally written in Swedish, although here they appear translated by Elin Lotsberg into her own west-Norwegian dialect. Her readings of the poems, with widely differing characters, provided the sonic and structural material for the entire composition.

Several phrases were transcribed using the ratio notation of Harry Partch, and then resynthesized in a "blown-up" manner. The overall time proportions in the piece also relate to proportions of speech and silence between the stanzas in a particular reading.

Many techniques of sound transformation were used, some familiar and some perhaps unorthodox, such as digital distortion (at most with as much as 90 dB), granulation, employing csound and a few routines written in C for this purpose, and phase vocoding. An original approach to phase vocoding has been developed at NoTam by Oyvind Hammer in his program Mammut. This program makes one single FFT of the whole sound, and makes possible some unusual kinds of transformations.

Here follows some fragments of the poems (English translation by the composer).

(1)

stirring the cup of tea in her room
at pace with the revolving music
- - -
discover a new way
of making yourself misunderstood:

silence and variations

she repeats herself
whilst

spreading out
like sweet marmelade

this tense atmosphere cannot be reproduced

with high enough fidelity
- - -

(2)
Behold the volcano, the lavae,
the living ash
- - -
Smoke of a flute is diffused
the frosen bay
- - -
Invisibly, we walk on the water
and draw portraits of each other
again and again

The material was recorded at the Norwegian Academy of Music in 1996, and the work was realised at NoTAM (Norwegian network for Technology, Acoustics and Music) in october 1997.

DRU (1997/98)

(for tape)

Fernando Iazzetta
Laboratório de Linguagens Sonoras
Comunicação e Semiótica - PUC-SP
iazzetta@exatas.pucsp.br

DRU was originally composed for a one-hour multimedia spectacle by Ivani Santana e Rachel Zuanon based on the work of Marcel Duchamp. Chance, probability and diversity are the base of this work which creates an interactive landscape by combining dance, image, music and the use of electronic technology. This piece is a condensed concert version and basically uses two types of sound sources: recorded voices and percussion instruments processed by different techniques such as cross-synthesis, filtering and granular synthesis, and; MIDI sequences produced by a careful combination of synthesized sounds controlled by an application written in MAX. DRU establishes a continual contrast between rhythmic material generated by random processes using the MAX environment and the more plastic material provided by the manipulation of recorded sounds. The result is an open "soundscape" inhabited by different sounds in constant interaction.

DRU foi composta para fazer parte de um espetáculo envolvendo dança, musica pré-gravada, projeção de vídeo e slides e iluminação. Alem disso, micro-cameras de video presas ao corpo das bailarinas geravam imagens em tempo-real, enquanto sensores de luz infra-vermelha dispostos no palco e na platéia controlavam o disparo de sons e imagens de maneira interativa. O espetáculo inspirado no trabalho de Marcel Duchamp tem 5 quadros (Rodas de Bicicleta, Ar de Paris, Etant Donnes, Jogo de Xadrez e Ready Made) com duração aproximada de uma hora. Essa versão abreviada para concerto (aproximadamente 8 minutos) e' quase que um resumo do material sonoro utilizado na versão integral. Basicamente existem três tipos de material sonoro bastante contrastantes na peca. O primeiro se apresenta logo no inicio, consiste em uma seqüência rítmica gerada através de processos randômicos criados no ambiente MAX. Uma batida forte constantemente deslocada sobre um pulso muito rápido seguida de pequenos motivos melódicos em quiálteras gera um movimento instável que oscila entre o regular e o aleatório. Os sons são cuidadosamente montados pela superposição de diversos timbres provenientes de 3 sintetizadores comerciais controlados via MIDI. O resultado tenta escapar da obviedade dos sons pré-fabricados criando um timbre composto de muitos outros timbres.

Um segundo material consiste de diversas amostras sonoras de vozes (uma cena urbana em Paris, a voz de um cantor de Tuva) e de sons percutidos que são transformados de diversas maneiras. A voz com os característicos multifônicos do canto de Tuva, por exemplo, e' apresentada em cross-sintese com o som de um didgeridoo, e os sons de um tamtam se

desenvolvem pela peca transformados por diversos tipos de filtragem (a maior parte delas realizadas no programa AudioSculpt) de modo a adquirir características bastante diferentes.

Finalmente, o terceiro material foi gerado através de síntese granular. Sao sons que passeiam por todo o espectro sonoro atingindo regiões limites no grave (50Hz) e agudo (15kHz). O resultado as vezes lembra o som produzido por um bando de muito grande de pássaros em vôo dando uma dimensão espacial mais ampla `a peca.

O efeito desejado e' o de criar um grande espaço sonoro, uma "soundscape" onde cohabitam diversos personagens acústicos que interagem durante todo o desenrolar da peca.

AtoContAto (1997)

(for tape, images, interactive tap shoes and performer)

Jônatas Manzolli [1], Artemis Moroni [2], Christiane Matallo [3]
Jonatas@nics.unicamp.br, artemis@ia.cti.br,
[1]Interdisciplinary Nucleus for Sound Studies (NICS) - University of Campinas (UNICAMP)
[2]Automation Institute - Technological Center for Informatics (CTI)
[3]Tap Dance Studio "Christiane Matallo"

As computer technology develops, the high-end computing environment no longer limits its applications. AtoContAto, Portuguese for act and contact, act with contact or act with touch intends to make human gestures close to sound as the dancer establishes a closer contact with music. Through a machine interface, a performer senses, touches and integrates music with dance. AtoContAto is a performing act. AtoContAto is based on a new gesture interface: a pair of tap shoes. Pizo-electric sensors were applied inside the taps, in the region underneath the toes and heel, and a cable terminated at that point. To simplify the electronic hardware, the total number of force sensors was limited to four, two per foot. These points are considered consistent with the dominant peaks of distribution of force along the base of the foot. Also, in this case, they supply enough amount of information. The sensors connected to an analog interface circuitry through a cable harness. There, the sensor signals are conditioned and digitized by a small microcontroller. The analog circuitry and microcontroller comprise a small module which is worn on the waist. The microcontroller translates the data into packets and sends them across a standard serial interface. The result is a MIDI Control Signal that can be plugged to several MIDI devices. The pizo-electric sensors underneath the performer shoes can be use to control sonic, light and image transformations. The junction of the dancer language with the sensor features results in a rich and unique mixture. The performer is transformed in a musical instrument, or "humanmatic device". The foot-mounted gesture interface enables a new sensory experience, strengthening interaction between sounds, rhythms and images. This new musical instrument enhances the relationship between dance and music. Here, cooperative or combined behaviors between human and machine create an emergent system. The result is a dance artwork in which a performer creates free movements, which produce changes in the sound material. The music is very much rhythmic. A blending of traditional tap rhythms mixed up into several different patterns. The sounds are supposed to be integrated to a collection of images. As with sounds, visual structures can be re-built during the performance interacting with the dancer's shadows, so that the dancer location on the stage changes the final result. The pizo-electric sensors underneath the performer shoes control sonic transformations. The panoply of rhythmic patterns allows a big variety of acoustic stimuli. The media - the interactive shoes - allows exploring the potential of this complexity. The result is a MIDI Control Signal that can be plugged to several MIDI devices. The junction of the dancer language with the sensor features results in a rich and unique mixture. The performer is transformed in a musical instrument, or "humanmatic device". In movement-based activities such as dance or sports, a performer's self-described movement orientation is closely related

to his or her level of performance achievement. As a practical application of this technology, we foresee devices that provide feedback closely correspond to internal sensations of movement, in order to assist a performer to evaluate and modify movements. Sports training and physical therapy are areas where a performer is engaged regularly in movement-based self-evaluation and movement modification. By tracking both locomotion and weight distribution we can search for combinations of transitions that might correspond to a particular movement performance that requires corrective attention. We envision a performer exercising a repertoire of movements while attending to a visual or auditory display controlled by those movements. A performer could listen to musical sequences and fine-tune the sounds by refining his or her corresponding movements. We foresee that this technology will have an application as an enhancement of existing devices for measuring physical performance. The foot sensing and gesture inference technology is still in an experimental stage. The continuous kinesthetic presence of a human in a computing interface is a powerful idea. For the intended context, the results obtained were very impressive. Observers were able to appreciate and understand the relationship between the actions of the performer and the corresponding musical transformations. The results enlightened the fact that we are not yet accustomed to such a bandwidth coupling between human and machine. Now, we are currently studying more complex patterns and the its basic properties, both in the method of their description by humans and in the construction of rules for recognition by machines.

Requiem per una veu perduda (1997)

Kyrie and Gloria

(for mezzo-soprano with effects and recorded electroacoustics)

Eduardo Reck Miranda
Department of Music
University of Glasgow, UK
http://website.lineone.net/~edandalex

Requiem per una veu perduda (*Requiem for a lost voice* in Catalan) is a piece for mezzo-soprano with effects (pitch shift, reverb, etc.) and recorded electroacoustic material (either on DAT or CD). The latter is fixed, that is, it does not 'change' during performance, but the soprano has some flexibility for synchronising her singing with this material. The piece was commissioned by Phonos/Pompeu Fabra University's Audiovisual Institute (IUA), Barcelona, and it was completed in July 1997. So far, it has been performed live in Cuba, Scotland and Spain, and broadcast in Argentina by Radio de la Cuidad de Buenos Aires. The Requiem has four movements of approximately four minutes each: *Kyrie*, *Gloria*, *Sanctus-Benedictus* and *Agnus Dei* (only the first two were featured at SBC&M due to time constraints).

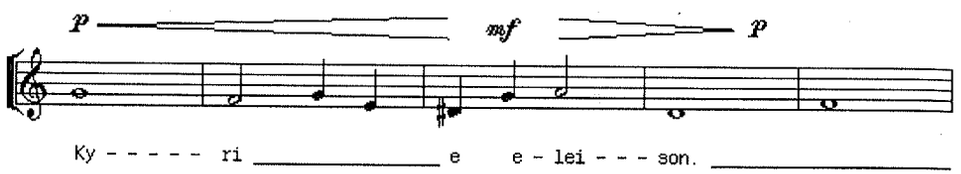
The electroacoustic material was produced using three types of synthesis techniques: (a) physical modelling, (b) spectral modelling and (c) granular synthesis [Miranda 1998]. For granular synthesis, I used *Chaosynth*, a software of my own design [Miranda 1995]; the *granular* sounds (sic) can be heard at various sections in the piece. Spectral modelling was used to modify either or both the form and the content of the spectrum of various sampled sounds, mostly from Catalan poetry readings. The systems used here were the SMS package (Serra 1997) and CDP's Phase Vocoder (Fishman 1997). But it is the physical modelling technique that is the main focus of this short essay, as it was used to synthesise the Latin lyrics of the *Gloria* and the speech-like sounds of the entire piece.

The pitch material for the mezzo-soprano part was defined based upon the analysis of the inner structure of the spectrum of vocal sounds. The relationship between the speech synthesis methods and the pitch material establishes the structural foundations of the piece.

The spectral contour of vocal-like sounds has the appearance of a pattern of 'hills and valleys'. The central frequencies, the amplitudes and the bandwidths of the peaks define the colour (or timbre) of a vocal sound. For example, the central frequency of the first formant of a vowel /a/ (as in the word "sanctus", in Latin), sung by a tenor is approximately 331.12 Hz whereas the value of the first formant of a vowel /o/ (as in the word "eleison", in Latin), is approximately 196 Hz. Each vowel sound is associated with a specific formant configuration and by changing the shape of our vocal tract we change the lower formants as we speak or sing. The three lowest formants are the most important in making vowels recognisable. In male adults, the first

formant can vary between 250 Hz and 1kHz, the second can vary between 600 Hz and 2.5 kHz and the third between 1.7 kHz and 3.5 kHz.

I am particularly interested in composing with vowel-like sounds, as I believe that our ability to recognise vowels is closely related to our ability to recognise timbre. I find that this phenomenon is a good starting point for the definition of a framework to compose a piece. The pitches and, to a certain extent, the whole harmonic structure of the Requiem were defined according to the values of the first three formants' central frequencies for the vowels /a/, /e/, /i/, /o/, and /u/, as they are pronounced in Latin, sung by male and female voices. After much research and experimentation, I defined a note-based formant system by matching the frequencies of musical notes to formant values (based upon the standard Western 12-tone equal temperament system, with a tuning reference of A4 = 440 Hz). These notes were organised into a number of different pitch-sets (e.g. F0, F1, F2 and F3 sets, a set of vowel /a/ values, etc.). Then, the individual sets were used to produce the musical passages for the mezzo-soprano part. Figure 1 illustrates a typical excerpt from the *Kyrie* using notes from the F0 pitch-set.



References

[Fishman 1997], Fishman, R., "The phase vocoder: theory and practice", *Organised Sound*, 2(2):127-145.

[Miranda 1995], Miranda, E. R., "Chaosynth: Um sistema que utiliza um automato celular para sintetizar particulas sonicas", *II SBC&M*, Brasil.

[Miranda 1998], Miranda, E. R., *Computer Sound Synthesis for the Electronic Musician*, Oxford (UK): Focal Press.

[Serra 1997], Serra, X., "Musical sound modeling with sinusoids plus noise", Roads, C. et al. (Editors), *Musical Signal Processing*, Lisse: Sweets & Zeitlinger, pp. 91-122.

Three Inconspicuous Settings (1998)

(for tape)

Aquiles Pantaleão
City University, London
A.Pantaleao@city.ac.uk

"Three Inconspicuous Settings" sums up a varied collection of personal references. Unrealised ideas kept for many years, objects and observations from my daily life and current compositional interests suddenly met in the formalisation of this piece. This explains the choice of materials and the overall design of the piece's structure. There is also a distinct environmental inspiration though clear references to nature and the use of environmental sounds are few, invariably disguised within more abstract contexts.

But perhaps, more than landscapes or environmental settings, the piece is concerned with mental states or modes of perception. In this sense, the first movement - contemplative and static in its nature - best represents the intention to avoid my usual notion of development as the creation of a 'no-goal' state of things was attempted. In this case no particular sense of direction is suggested, and as the movement slowly unfolds, materials freely flow back and forth while avoiding sudden surges and dramatic impacts or conflicts. As such, the piece tries to surface the subjective responses drawn from the observer rather than a meaningful rendering of nature itself. Hopefully this is reinforced throughout the piece by the recurrence of similar events and morphologies and the ubiquitous presence of pitched materials coming in and out of textures.

The movements were called 'inconspicuous' as they may be undistinguished from one another due to their possible likelyhood. And also because they form what at first I considered to be a lesser or unimportant piece. But thankfully I changed my mind on that.

And before I forget... the movements are individually named as Outside (0'-4'42"), Inwards (4'48"-8'35") and Outbound (8'35"-12'24").

The Triangle of Uncertainty (1995-1996)

Suite for maritime landscapes
(Musique concret for sound installation,
produced as part of l'Imaginaire Irlandais 1996)

Nocturnes 2 and 3

Cécile Le Prado
IRCAM, Paris
Cecile.Le.prado@ircam.fr

Concept

Sailors and seafarers find their bearings at sea by mean of natural points of reference located along the coast. These points, for example church spires, hills, water towers or lighthouses that generally stand out from the rest of the coastline, are called amers (seamarks or landmarks). All you have to do is to identify three such landmarks in complementary directions so as to be able to construct a triangle which inevitably contains your ship. This triangle drawn on the navigation map is called the "Triangle of Uncertainty".

The sound installation principle Le triangle d'incertitude takes up the principle of triangular navigation substituting these visual seamarks with acoustic landmarks -also information elements used in navigation-that can be recognised by listening carefully. Lighthouses, buoys, ship'radio and many other technical facilities warn the sailor of hazards or obstacles. This installation project is concerned with constructing a triangle of uncertainty on the ground of a fictive, virtual space on the basis of sound recordings made at the following locations: the southern tip of Ireland (Fastnet Rock), the western edge of France (Brittany), and the westernmost point of Spain (Cap Finisterre, Galicia). In essence, the installation refers to the position of sounds in space, constantly chopping and changing between orientation and uncertainty.

Realisation

The recordings play an important role for the conception of the work and are made by Cécile Le Prado. Both preselected sounds are recorded, such as a lighthouse foghorn or a wistle of a buoy out at sea, as well as sounds discovered coincidentally during recording. The sounds are recorded because of their extraordinary, innate acoustic and musical qualities which gives the recording a very specific, irreversible three dimensions image. One example is water gushing through a hole in the pier of Malpica in Spain. Owing to its musical aspects, this sound resembles a reworked studio sound, although maintaining its typical arbour quality. The history of the recording also plays an essential role during composition work in studio, for although the ear is indeed stimulated at this particular moment, certain recollections of the circumstances,

visual and other impressions will actually decide on what nature the installation will assume at a later point.

In the studio the recorded sounds are digitally sorted and cut, and some of them are reworked before taking their place in a sort of storyboard. This reworking is often triggered by a harmonic or temporal quality (to widen the concept of rhythm), a quality that already existed when the original recording was made. By simulating an acoustic space in real time and by selecting the path these sounds sources take in this space, what is created is an"écriture de la spatialisation", writing for spatial recording, as an element of composition. It is gradually refined by repeated listening and modification.

This work was created with the aid of the Spat ® spatialisator. This newly developed software by Ircam and the Atelier Espaces nouveaux is a virtual acoustic processor which is essentially based on a perceptive analysis of space. With the aid of the Spat ®, this construction simulated in studio can be adapted to meet the requirements of various playback sites. The final stage of realisation consists in integrating the installation implemented in the studio into an outdoor or interior setting. The interplay between the composition, the "natural" acoustic environment and the acoustics of the installation site is always very interesting. Certain aspects of the studio composition vanish, while others become more important. The sites develops and gradually takes on the form of the suggestion made in the studio, particularly depending on the time of day, weather conditions, etc. in case of an outside installation. The interaction of a given site and installation determines a specific combination of restrictions and options. In this sense, the "Triangle of Uncertainty" is extremely constraining. It is heard under very precise conditions. The loud speakers have to be equidistant to each others and set up in a circle. The ideal space of perception, in which the spatial effects can actually be perceived, is limited to a relatively small area within the circle.

TERRA (1997–98)

(for tape)

Jean-Claude Risset
Laboratoire de Mécanique et d'Acoustique of CNRS, Marseilles
jcrisset@alphalma.cnrs-mrs.fr

Terra is a section of Elementa, a work commissionned by the French Ministry of Culture for the fiftieth anniversary of musique concrète and realized at INA-GRM, Paris.

The piece was realized with Pro Tools and GRM Tools. It includes sounds synthesized with the Music V program and the Synclavier synthesizer, but a substantial proportion of the sound material consists of recordings. The nature of the sound sources is not hidden: the composition relies upon their connotations and their symbolic implications. The recorded sounds are weaved into figures, phrases, developments and sections: a compositional processing, careful to preserve the autonomy of organic sound objects and their dynamics of flux, duration and energy.

Terra evokes our vital sphere, with the mineral, vegetal and animal order. The solid state of matter is illustrated through its different forms of vibration: rolling, friction, percussion, creaking, plucking, explosion ... After a long expectancy and a passacaglia of pebbles, everything is rocked: in an avalanche, even earth and stones flow.

Outermost (1998)

(Film/musical composition)

Stephanie Maxwell, Filmmaker,
Associate Professor, Film/Video/Animation Deparment, School of Photographic Arts, Rochester Institute of Technology,
sampph@rit.edu

Allan Schindler, Composer
Professor of Composition, Director, Eastman Computer Music Center, Eastman School of Music, University of Rochester
allan@esm.rochester.edu

This collaboratively conceived and realized work features a uniquely close integration of visual and musical imagery and gestures, not merely in terms of general esthetic qualities and synchronized "hits" but rather by means of continuously evolving and changing relationships and patterns between these two elements. We conceived and view Outermost as a film/musical composition, rather than a film with a musical soundtrack. Underlying formal concepts, abstract imagery, and qualities of movement are mirrored in both the visual animation and in musical motifs, but not always synchronously. Rather, the visual and music patterns "chase" each other, "dance" intertwine, pull apart and come back together in long swirling arcs and patterns. The music underlines or makes more explicit certain recurrences and arrival points in the animation, and vice versa. As a result, we believe that the aggregate shape, emphasis and impact of the work are quite different from what one would experience from either the animation or music alone.

The animated imagery was originally hand painted frame-by-frame on 35mm motion picture film stock using paints and markers. This method of inscribing designs directly on the film itself, called direct animation reproduces the individual physical impulses of the artist by bypassing the camera intermediary. Films produced by direct animation techniques visually and conceptually reflect the individual artist's personal relationship with his or her materials and tools, and unique methods of creating cinematic imagery. This art form is very physical, expressing bodily impulses, vibrations from fingertips, changes in applied pressure, externalized internal rhythms and aesthetic preference. The frame-by-frame direct animation processes used in Outermost involve air brushing, stencilling techniques, direct application on the film of marker and water color pigments, etching directly into the film's emulsion, and "negative painting" meaning that the colors painted originally on the film were selected for the alternate colors that would result from their eventual computer color shifting (i.e., colorizing).

The original 35mm film imagery was then trans-formed by video and digital processing (Rank Telecine and Copernicus Interfacer color shifting, and orientation, direction and framing manipulations of the original moving designs) and transferred onto Beta SP videotape. This

new imagery was then subjected to a rigorous editing. In the four minutes of imagery in this film there are 157 edits (or cuts). A deliberate part of the overall visual experiment in this film was to "activate" or intensify the original animation through creative editing strategies to produce unusual movement characteristics and a unique cinematic experience of colorful, painterly abstract moving forms.

The music was realized by means of software procedures on SGI 02 and Indy computer systems. Many of the sound sources were derived from computer analysis, resynthesis and transformations of Western, African and Asian acoustic sounds. Principal software employed included Csound (Vercoe), Score-11 (Brinkman), rt (Lansky), mix (Hammer), SMS (Serra), linear prediction software, simulated spatial movement and localization programs, and algorithmic compositional programs, written by the composer, that generate note or event streams based on probabilities, permutations or transformations.

"Dissequentia" (1997)

(for tape)

Agostino Di Scipio
Centro di Sonologia Computazionale
University of Padova, Italy
LMS@aquila.infn.it

This work in sounds and voices is an attempt at "deconstructing" a poem by Edoardo Sanguineti, titled Sequentia, which was in turn inspired by and dedicated to Luciano Berio's virtuosic solo instrument works globally called Sequenze. The attempt was to destroy the musicality of Sanguineti's poetry on its own ground, namely on the ground of the purely sonorous (or purely "musical") appeal of his words, and to let a different meaning emerge from the very same words we read in the original Sanguineti. This experiment "in words" (we also hear a voice whispering "parole!", i.e. "words" in Italian, a quote from Berio's electroacoustic historical "Visage", from the '60ies), extends to an experiment with the sound of the paper sheets upon which words are usually printed: the sound of skimming through books and magazine, ripping off the pages, etc., are granulated, stretched, heavily transformed in various ways, up to the point w here they become similar to voices whispering the Sanguineti's lines. There are 10 lines in all, each of which is introduced by a voice enumerating the 10 sections in the piece (after the 3rd section, however, the speaker gets rather confused and jumps ahead to the 6th, then gets back from the 8th to the 4th, and so on, until finally she tries to utter "ten" and "eleven" at once...).

Thanks are due to Eva Martelli for lending her voice.

The work was produced by the composer programming the Kyma computer music workstation at Laboratorio Musica e S onologia, L'Aquila.

WHAT HAPPENS BENEATH THE BED WHILE JANIS SLEEPS? (1997)

(for tape)

Rodolfo Coelho De Souza
University of Texas, Austin
rcoelho@mail.utexas.edu

During the hot summer it is hard to sleep.
The large corridors are empty, only ghosts live there.
Where is Janis now?
When she sleeps something moves under her bed.
Call them dwellings of the underground,
incubus that came to sing lullabies to Janis.

This piece was created in the laboratories of the University of Texas at Austin during a doctoral program in electronic composition, under the supervision of Russell Pinkston, D.M.A. , and supported by a scholarship from the CNPq.

The development of the piece involved many different digital techniques according to the aimed results. Recorded samples were required to undergo multiple processing in programs like Sound Designer, Dolson and Csound, or digital devices like the Harmonizer and the mixer Yamaha O2R. Some synthesized materials were generated in Csound. The final edition was made using the program Pro-Tools in a Macintosh platform.

The digital processing of the human voice is the central element of the piece. From abstract transformations of human speech, the introduction builds a woman's shout. A shout that evokes Munch's painting. This recurrent shout generates, by transformation and association, all the episodes of the piece, drawing a surrealistic pseudo-narrative, like the manifest content of a Freudian dream without the interpretation of its hidden meaning.

This language of free associations of referential materials requires techniques of distortion to reach the fundamental iconic level. The transformation of referential materials in abstract iconic sounds is called by Schaeffer's aesthetic of concrete music the *acousmatique* veil. These abstract sounds can be linked and related again to new referential sounds allowing the composer to build the chain of associations. The human voice when transformed becomes an animal's voice, or a submarine creature, or a flying monster. But the voice is nothing but air passing through the throat, human wind. This wind can be transformed in a tempest or a thunder. Another transformation and the wind becomes a herd of running horses. Transformed, horses become human footsteps and transformed again, footsteps are machines clanging: a train, Schaeffer's train, Gobeil's train, all trains of our experience.

When, by metonymy, the train's whistle blows, it is not a whistle anymore. By convolution it became the voice singing and we are back to the human element. Each transformation is a concrete metaphor or a concrete metonymy, the two basic processes that Jakobson considers the fundamental linguistic operations of poetry. But Schaeffer's concepts also remember us that what allows the articulation of all these transformations are the abstract iconic relations that we find in these sound objects.

Peel (1997)

(for tape)

Pete Stollery
Northern College, Aberdeen ,Scotland
p.stollery@norcol.ac.uk

Peel continues recent concerns in my music to investigate the musical and structural potential which lies behind sounds which are familiar to us. 'Peeling' back the outer layers of these sounds reveals new unfamiliar sounds which, when juxtaposed with the familiar sounds, create surreal soundscapes. Sometimes the peeling is violent, sometimes gentle. The boundaries between the real world and unreal worlds become blurred - eventually we begin to recognise the unknown.

Peel was commissioned by BEAST with funds provided by West Midland Arts. It was realised in the Electroacoustic Music Studios at Northern College, Aberdeen and at the University of Birmingham and was also produced using the One Voice Scottish Composers Resource, with the support of the Scottish Arts Council and the UK National Lottery.

Peel was composed using recorded sounds (natural and synthetic) which were edited and processed within a ProTools environment using software and plug ins such as SoundHack, GRMTools, DVerb, Hyperprism and Audiosculpt.

Segmentation fault Beta 1.1 (1996-97)

(for piano and live electronics)

Marco Trevisani
New York University
marco.trevisani@nyu.edu

All the compositions, written for the Songo Be'nd Project are part of the same compositional work, and Segmentation Fault is one of those. They are written to be played with instruments and live electronics, in various and always different combinations, and they are mainly based on rhythmical improvisations. Independently from the composition of the ensemble, it always takes the name of Songo Be'nd, from the cuban poet Nicolas Guillen. The other constant element is the prepared piano, as well as some tools for signal processing (to process the piano sound) and a computer (for triggering and play in real time, pre-processed sounds). The attempt is to incorporate polyrhythmic elements not found in Western music into our own musical style. The rhythm becomes the dominant element. Also the piano is prepared in a way where its harmonic characteristics become hidden and its rhythmical aspects are taken into a predominant relevance. Even if I write, for all compositions, the main structure and the pre-processed sounds, the final interpretation of the piece is always the result of a very intense collaboration with the musicians involved in the specific session. In Segmentation Fault Beta 1.1 also the structure is the result of the collaboration with Michael Edwards. The computer, using custom-written software by him, is used to trigger and mix pre-processed sound material. The original program runs on an SGI or a Linux plattform but other programs are available for different platforms to play and trigger pre processed sound files. In case of necessity, a prerecorded dat tape with all the pre-processed sounds can be played, while instruments are playing. Any other musician can be added, depending on the circumstances, and it could be drum/percussion and/or electric guitar and/or saxophone and/or trumpet.

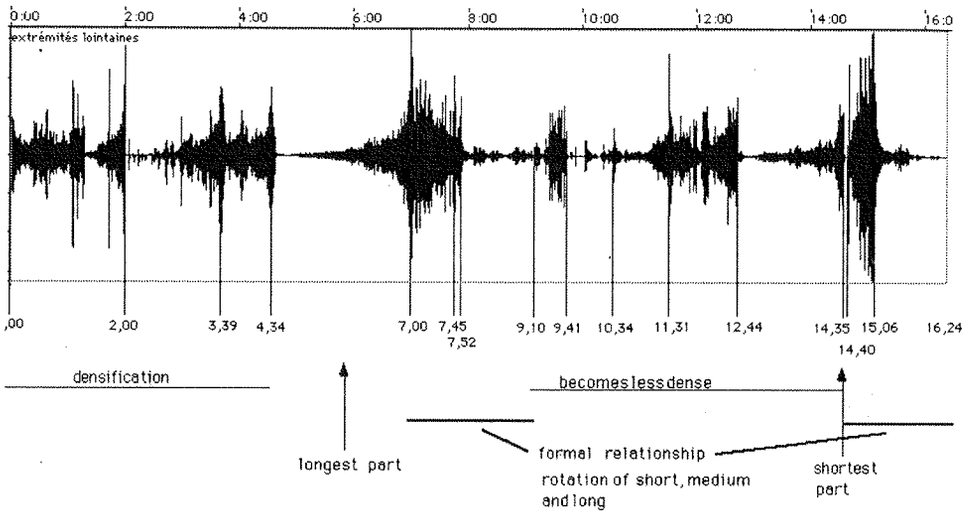
extrémités lointaines (1998)

for 8 channel tape
comissioned by Ina-GRM 1998, production: GRM

Hans Tutschku
IRCAM, Paris
hans.tutschku@ircam.fr

"extrémités lointaines" uses recordings I made during a four-week concert tour in Asia in the summer of 1997. In Singapore, Indonesia, the Philippines and Thailand, I recorded the sounds of big cities, the music of churches and temples, and the songs of children.

These sounds were then processed in the studio, using mainly real-time granular synthesis. This synthesis was controlled either by compositional rules set up in MAX, or gesturally - i.e., by "playing" a faderbox and listening to the results. In this way, all the recordings were cut into tiny pieces and reassembled, establishing many new relationships among the sources. Furthermore, filtering and cross synthesis of the sounds' spectrums allowed me to achieve continuous changes between different sources.



For many years, I have been integrating vocal and instrumental sounds in my electroacoustic compositions, while maintaining the cultural context and sound environment of these sources. "extrémités lointaines" continues and extends this process, presenting the sources within the cultural context of four different countries.

The piece is structured in 15 parts, describing different places and atmospheres. The intensity of impressions I experienced during the trip lead to a very dense compositional structure..

tutorial

Introdução a Síntese e Processamento Digital de Áudio por Computador

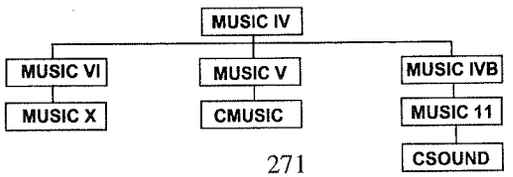
Victor E P Lazzarini
Núcleo de Música Contemporânea
Departamento de Arte
Universidade Estadual de Londrina

Resumo

Este tutorial tem o objetivo de apresentar aos iniciantes em Computer Music os elementos básicos de síntese e processamento de áudio. A abordagem será direcionada para a prática composicional com o uso do programa Audio Workshop 1.3 e do compilador sonoro Csound. O programa do tutorial tratará inicialmente do histórico e dos elementos básicos do funcionamento dos programas que serão utilizados. Em seguida serão estudadas as técnicas básicas de síntese e processamento de áudio com a utilização do computador em exemplos práticos.

1. As linguagens de Computer Music

Os primeiras tentativas de uso do computador como ferramenta para a criação de sons através de síntese datam da metade dos anos 50. Nesse período, desenvolveram-se , na divisão de pesquisa do Bell Telephone Laboratories em New Jersey (EUA), alguns estudos voltados para a digitalização de sinais de áudio para fins de uso em telefonia. A complexidade envolvida nesse processo levou os pesquisadores a utilizarem-se do computador como um auxiliar no desenvolvimento das pesquisas. Apesar dos grandes obstáculos técnicos a serem transpostos, foi observada a possiblilidade futura para o desenvolvimento de sistemas de áudio digital de qualidade. Um pesquisador trabalhando nesse laboratório, Max Mathews, desenvolveu os primeiros programas experimentais para a síntese direta de áudio digital por computador, chamados MUSIC I (1957) e MUSIC II (1958). Eles foram escritos para o computador IBM 704, de construção baseada em válvulas, extremamente limitado, e produziam sons muito simples de ondas triangulares. Com o aparecimento de um computador com maior capacidade, o IBM 7094, em 1959, Mathews desenvolveu um programa de síntese um pouco mais completo, o MUSIC III, terminado em 1960. Neste momento, outros pesquisadores envolveram-se no desenvolvimento da pesquisa em Computer Music (CM), e o programa



seguinte da série desenvolvida por Mathews, MUSIC IV, foi tomado como o modelo básico de uma família de programas para CM, que são o que se chamou de *computer music languages*, ou também de *sound compilers*, compiladores sonoros. O esquema abaixo mostra a relação entre os diversos sistemas de síntese e processamento de áudio derivados do MUSIC IV:

A característica comum desses programas é o uso de uma série de definições textuais que vão definir a maneira como o computador vai gerar o sinal de áudio digital. Pode-se falar então de *instrumentos* no computador, que são algoritmos que realizam um evento musical. O usuário define, por meio de uma sintaxe particular, os *instrumentos* de uma *orquestra,* para sintetizar certo tipo de som, além de definir uma *partitura,* para ser executada pelo programa. Os *instrumentos* definidos pelas linguagens são compostos de elementos menores, chamados de unit generators, ou unidades geradoras (UG). As UG's geralmente possuem uma série de inputs, produzindo um ou mais outputs. A construção de instrumentos no computador utilizando uma linguagem de CM é baseada no arranjo e interconexão de UG's para se conseguir o algoritmo de síntese pretendido. Geralmente, no processo de definição de um instrumento, faz-se uso de diagramas de fluxo de sinal, onde as UG's são definidas por meio de figuras, as conexões, por meio de linhas, e as operações aritméticas simples são representadas pelos seus sinais habituais. Como veremos, algumas UG's, como os osciladores, são usualmente representadas por determinadas figuras padrão.

2. A sintaxe da linguagem csound

Csound tornou-se a mais importante linguagem de CM da família MUSIC IV, pelo seu grande desenvolvimento, acessibilidade e disponibilidade na maioria das plataformas computacionais. Examinaremos aqui os elementos básicos de sua sintaxe. Csound funciona com dois arquivos de código, a *orchestra* e a *score*. O primeiro arquivo deve conter as especificações dos instrumentos em termos de UG's e fluxo de sinal, enquanto o segundo contém informações relativas à performance desses instrumentos: tempo de início do som, duração e outros parâmetros definidos em cada instrumento, além das definições de tabelas que serão usadas por algumas UG's. Os dois arquivos são passados ao programa como argumentos em uma linha de comando, juntamente com outras opções. Em alguns ambientes gráficos e front-ends, o processo é similar, diferindo apenas a aparência gráfica do programa. O programa pode então gerar um arquivo de áudio em um certo formato, ou então, em certos sistemas, escrever diretamente para a placa DAC, realizando síntese em tempo real.

2.1. Orchestra

O código do arquivo *orchestra*, geralmente com nome *.orc, pode ser separado em cabeçalho e definições de instrumentos. No cabeçalho são definidos alguns parâmetros que serão usados no processo de síntese, que são: sampling rate (sr), que é a freqüência de amostragem do sinal de áudio[1], em Hz; control rate (kr), que é a freqüência de amostragem do sinal de controle, em Hz; número de samples em um período de controle (ksmps), equivalente a sr/kr; e, finalmente, número de canais de saída de áudio (nchnls). Assim, um exemplo de cabeçalho poderia ser:


```
sr=44100
kr=4410
ksmps=10
nchnls=1
```

onde o som sintetizado teria 44.1 K Hz de freqüência de amostragem e seria mono. O sinal de controle usado no processo de síntese teria freqüência de 4410 Hz e haveria 10 samples em cada período de controle. Como veremos, as variáveis do tipo **k** serão atualizadas uma vez por período de controle, enquanto as variáveis do tipo **a** serão atualizadas a sr vezes por segundo.

As definições de instrumento começam pela declaração *instrN*, onde *N* é o número arbitrário que designará o instrumento para o programa. A declaração *endin* conclue a definição de um instrumento, *instrN* e *endin* devem sempre estar casadas ao longo do código de uma *orchestra*. Entre essas duas declarações coloca-se a definição do instrumento, em termos de variáveis, parâmetros para as UG's, opcodes (que são os identificadores das UG's), e outras declarações (como por exemplo as de controle de fluxo, conversões e etc.).

As variáveis utilizadas na definição do instrumento podem ser de quatro tipos básicos: **p***n*, parâmetros passados à *orchestra* provenientes do arquivo *score*; **i***var,* variáveis escalares que tomam valor apenas no tempo de inicialização, geralmente uma vez a cada *nota* sintetizada; **k***var*, variáveis escalares que são atualizadas kr vezes por segundo; e **a***var*, vetores usados para guardar sinais de áudio, amostrados sr vezes por segundo. O entendimento do uso dessas variáveis é muito importante para o bom design de um instrumento. Usa-se as variáveis do tipo **i** para armazenar valores que serão usados somente uma vez no processo de síntese de um som, por exemplo, uma freqüência fixa. As variáveis do tipo **k** são usadas para parâmetros que evoluem lentamente no tempo, assim economizando operações, melhorando a performance do instrumento. Os sinais representados pela letra **a** são geralmente reservados para os sinais de áudio. A definição de um tipo de variável é feita iniciando-se o nome da variável pelo tipo que ela representa: ksinal, ifreq, aoutput, etc.. O exemplo abaixo mostra o uso das variáveis:

```
sr=44100
kr=4410
ksmps=10
nchnls=1

instr1          ; comentários começam com ";" e vão até o fim da linha
iamp = p4       ; p4 é passado do arquivo score
ifreq = p5      ; p5 idem
kamp  line      0, p3, iamp  ; kamp recebe o sinal de uma UG (chamada line) que faz o sinal variar
                             ; linearmente
                             ; de 0 a iamp em p3 segundos (p3 é a duração deste som, passada
da                           ; score)
aout    oscil   kamp, ifreq, 1 ; aout recebe o sinal de outra UG, um oscilador, que gera um certo
                               ; tipo de onda com amplitude kamp e freqüência ifreq. A constante
```

```
                ; 1 como último parâmetro refere-se a uma tabela de função n. 1,
                ;  definida na score, contendo um ciclo de onda de uma form
                ; a arbitrária, que será usada pelo oscilador no processo de síntese.
        out     aout            ; o sinal aout é colocado na saída do instrumento
endin
```

O instrumento acima gerará um som cuja freqüência é definida pelo parâmetro p5 da *score* e com uma amplitude que vai de 0 ao valor definido por p4 também na *score*. A forma de onda usada será definida também na *score*, pela tabela de função n.1.

2.2. Score

O arquivo score é composto por uma série de declarações que podem ser de vários tipos: f, para tabelas de funções; i, para notas a serem executadas por instrumentos; a, para avanço sem produção de som; t, para a definição da medida de tempos usada; s, para a separação da score em secções; e, por fim, a declaração e, usada para concluir a score. A forma das declarações é sempre a mesma: começa com a letra identificadora (opcode) f, i, a, t, s ou e, seguida de uma série de campos de parâmetros, pfields, p1, p2, p3, p4...pN, separados por espaços. Examinaremos a seguir os tipos mais importantes de declarações da score, f e i.

As declarações do tipo f definem as tabelas de função que vão ser usadas pelos instrumentos da orchestra. Elas chamam as subrotinas GEN para colocar valores em uma tabela de um certo tamanho. Os campos de parâmetros para este tipo de declaração são assim definidos:

p1 número de identificação da tabela
p2 instante de geração (geralmente 0, gerando a tabela no início do processo de síntese)
p3 tamanho da tabela (quer dizer o número de posições do array), que deve ser uma potência de 2, ou potência de dois mais um ($2^n + 1$)
p4 número identificador do tipo de subrotina GEN que vai ser chamada para a criação da tabela, definindo o tipo de tabela a ser criada (p. ex. GEN 10 cria um ciclo de uma onda periódica com um certo número de harmônicos)
p5 - pN parâmetros cujo significado é determinado pelo tipo de GEN chamada

Exemplo:

```
; comentários começam com um ";"
; p1=ID  p2=instante de geração  p3=tamanho     p4 =tipo de GEN     p5=usado pela GEN
f 1      0                      1024           10                  1
```

define que a tabela 1, de 1024 posições, será preenchida por um ciclo de onda com um harmônico, na frequência fundamental. Neste caso, a GEN 10 toma os parâmetros p5 a pN como os pesos (amplitudes) individuais dos harmônicos em ordem ascendente da onda a ser gerada. Se aqui foi apenas definido um harmônico de amplitude 1, teremos uma onda do tipo senoidal preenchendo a tabela. Esta tabela poderia então ser usada pelo instrumento definido anteriormente, para a geração de um som senoidal.

As declarações do tipo i chamam um certo instrumento a tocar a partir de um certo instante, por uma duração de tempo e com alguns parâmetros (opcionais) definidos pelo usuário. Três campos de parâmetros, p1, p2 e p3, são reservados, com significação pré-definida, mas os outros parâmetros podem ser livremente utilizados:

p1 número do instrumento a ser utilizado
p2 instante de início do som
p3 tempo de duração
p4 - pN parâmetros cujo significado pode ser determinado pelo usuário.

Exemplo:

```
;p1=instr  p2=início  p3=duração p4=amplitude p5=frequência
i 1        0         5          16000        440
```

define que o instrumento 1 deverá estar ativo a partir do tempo 0, por 5 batidas. Define-se também que o parâmetro p4 equivale à amplitude e p5, à freqüência. Isso foi definido anteriormente em nosso instrumento da secção 2.1, iamp=p4 e ifreq=p5, e estes campos de parâmetros serão utilizados por este instrumento no processo de síntese. Se omitíssemos estes campos, o compilador geraria uma mensagem de erro, dizendo que os parâmetros p4 e p5 requisitados pelo instrumento não estão presentes na declaração i. Batidas (beats), é a unidade de tempo arbitrária utilizada em todas as declarações da score, e seu valor default é 60 batidas/minuto, equivalendo a segundos, que pode ser alterado utilizando-se uma declaração do tipo t.

Como exemplo final, uma score completa que tocará uma sequência de notas, com um som senoidal, no instrumento definido na secção 2., com diferentes amplitudes. Com relação a essas, note-se que,se estamos criando um som com precisão de 16 bits, a amplitude deverá ficar entre 0 e 32767, para não causar clipping (se isto acontecer o compilador informará o número de samples out of range). A declaração e fecha a score:

```
;                                  tamanho     GEN         usado pela GEN
f 1       0                        1024        10          1
;  .    início   duração   amp     freq
i 1       0        1        10000   220
i 1       1        2        16000   330
i 1       3        1.2      24000   440
e
```

2.3. Compilando o som

Após a definição da *orchestra* e da *score*, armazenadas em dois arquivos, digamos, respectivamente *sine.orc* e *sine.sco*, devemos utilizar o comando **csound** para compilar o som desejado em um arquivo de áudio de algum formato. A forma do comando é a seguinte:

csound orchestra.orc score.sco [opções]

onde opções incluem uma série de flags controlando o tipo de output, o formato de arquivos e algumas outras definições do processo de síntese. Se o comando é dado sem argumentos, uma mensagem de uso é impressa na tela do computador, com o sumário das opções. A seguinte linha de comando utiliza a *orchestra* e a *score* definidas nas secções 2.1 e 2.2, para gerar um arquivo do formato RIFF-Wave, 16-bit (default), *sine.wav* no diretório de áudio (definido pela váriavel de sistema SFDIR), se definido, ou então no diretório presente:

csound sine.orc sine.sco -W -osine.wav

3. O programa Audio Workshop

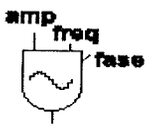
Como opção para o estudo da síntese e do processamento de áudio, existem vários ambientes gráficos que facilitam o uso do computador como gerador de sons. Dentre eles, o programa Audio Workshop (audioworks), desenvolvido pelo autor, apresenta os algoritmos básicos em uma forma gráfica, além de disponibilizar o display gráfico do som gerado e possibilitar a performance do arquivo de áudio gerado pelo programa. O funcionamento do programa é dividido em duas secções, síntese e processamento. O usuário apenas escolhe, em um menu de opções, o tipo de síntese/processamento desejado, e preenche os parâmetros requeridos. O programa então produz o arquivo de áudio e desenha um gráfico amplitude por tempo do som gerado. Pela sua facilidade de uso, o programa pode ser empregado no estudo deste tutorial, para a aplicação dos processos de síntese. Maiores detalhes sobre o program podem ser encontrados em (Lazzarini, 1998).

4. Elementos básicos de síntese digital

Como se falou anteriormente, podemos definir os algoritmos de síntese e processamento de áudio por meio de UG's. As UG's mais básicas para qualquer algoritmo são os osciladores, os geradores de funções temporais (geradores de envelope). Com estas UG's e algumas operações aritméticas, como soma e multiplicação, pode se criar uma série de algorítmos úteis para a síntese digital de áudio.

4.1. O oscilador

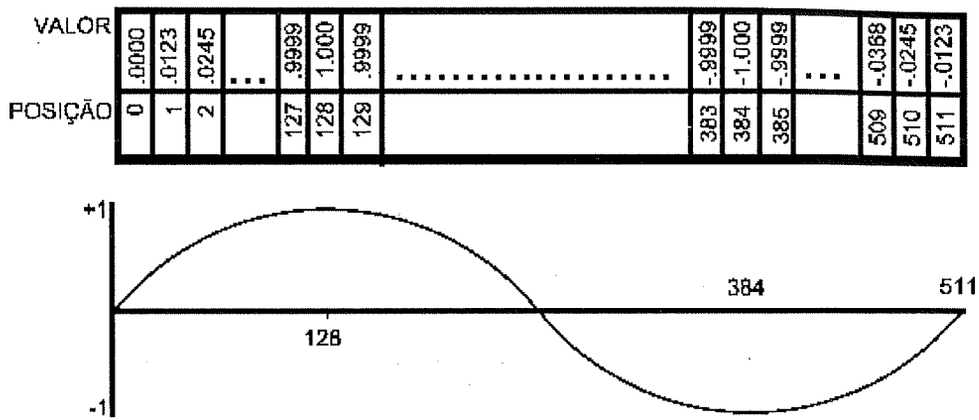
O oscilador é talvez a UG mais fundamental para a criação de algoritmos de síntese no computador. Ele gera uma onda periódica de forma arbitrária, com controles de amplitude, frequência, desvio de fase e tipo de onda utilizada. É comum o uso do símbolo mostrado abaixo, em digramas de fluxo de sinal, para representar um oscilador , suas entradas de controle e sua saída:



onde amp é o pico absoluto de amplitude do sinal, freq, freqüência, geralmente definida em Hz, e fase, o desvio inicial da fase, que também geralmente é de definição opcional. A forma da onda utilizada geralmente é estilizada no interior da figura (neste caso, uma senóide). Um oscilador poderia ser construído apenas se especificando uma função de tempo de algum tipo, para isso o oscilador teria que efetuar uma cálculo matemático para a geração de cada sample de áudio, o que seria muito ineficiente. O método mais eficaz de se implementar um oscilador é aquele que se utiliza de um ciclo de onda previamente calculado e armazenado em um array com um certo número de posições, chamada tabela ou tabela de onda. Uma vez calculada a tabela, o oscilador pode gerar os valores de saída apenas retirando-os da tabela em certas posições, e multiplicando-os pelo valor da amplitude. Como exemplo, o seguinte fragmento de código em C++ cria uma tabela contendo um ciclo de senóide:

```
const double DoisPI = 2*3.14159265;
float* tabela = new float[512];
for (int n = 0; n < 512; n++)
     tabela[n] = (float) sin(DoisPI*n/512);
```

A tabela é normalizada, isto é contém valores entre 0 e 1. A figura abaixo mostra o resultado de tal operação:



O algoritmo do oscilador guarda um valor de fase, que indica a posição corrente na tabela, de onde um valor é retirado e multiplicado pela amplitude, representado a saída do oscilador.A cada sample[2], o algoritmo adiciona a esse valor uma quantidade, chamada *incremento* (sampling increment), proporcional à freqüência, e esta soma é usada para encontrar a nova posição na tabela para o próximo sample. Se o valor da fase exceder o tamanho da tabela, uma operação de módulo é realizada, e assim o processo torna-se circular.

Dois tipos de osciladores são encontrados: de freqüência de amostragem (sampling rate, sr) fixa e de sr variável. O tipo mais comum é aquele de freqüência de amostragem fixa. Para examinarmos este tipo de oscilador, considere-se a tabela acima, de 512 posições contendo 1

:o de onda, onde existem, portanto, 512 samples em um ciclo. Com uma sr de 44100Hz, a freq. do oscilador (com incremento = 1) seria de 44100/512 = 86.13 Hz. Para se obter um som uma oitava acima, o algoritmo pode ser programado para obter valores a cada duas posições (incremento = 2). Pelo fato do oscilador ler a tabela duas vezes mais rapidamente, cada ciclo possue a metade do número de samples do caso anterior (256), e a freq. oscilador seria 44100/256 = 172.26 Hz. Assim temos a relação: incremento = N(freq/freq de amostragem), onde N é o número de posições da tabela.

O fragmento de código abaixo implementa um oscilador simples para ler a tabela apresentada acima. Trata-se de um algoritmo um pouco limitado, usado apenas como exemplo:

```
float amp = 16000.f, fr = 440.f, sr = 44100.f,
          dur = 1.f, saida;
float indice = 0.f, incremento = 512*(fr/sr);
  for (int n = 0; n < dur*sr; n++){
        saida  = amp*tabela[(int)indice%512];
        indice += incremento;
                                  }
```

O índice acima representa o valor da fase, que é usado pelo algoritmo para obter a posição corrente na tabela. Muitas vezes o incremento não vai ser um valor inteiro, como no exemplo acima, onde será 5.108. Nesse caso o algoritmo encontrará uma posição que será intermediária entre dois valores da tabela. O algoritmo acima descarta a parte fracional do índice, fazendo o processo chamado de truncagem. O outro modo que se utiliza é o de interpolação entre dois valores adjacentes na tabela, encontrando um valor intermediário. Este método é mais preciso, mas envolve mais operações para o cálculo de um sample. As imprecisões causadas pelos métodos citados vão resultar em uma forma de ruído que é adicionado ao sinal. A relação sinal-ruído (s/r), em dB, desses métodos é relacionada com o tamanho da tabela usada, quanto maior a tabela melhor a relação s/r. Para truncagem, a relação é aproximadamente $6(\log_s N - 2)$ dB, e para interpolação, $12(\log_s N - 1)$ dB, onde N é o número de posições da tabela usada. A relação s/r para o algoritmo mostrado acima então seria de 48 dB. Se interpolação tivesse sido usada, a relação s/r seria 96 dB.

4.2. Gerando tabelas de onda

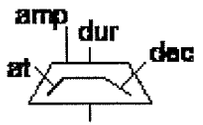
Para se gerar tabelas de ondas, em linguagens como csound, utiliza-se as subrotinas GEN que possam criar ondas periódicas. As GEN's 9, 10 e 19 existem para isso. Como exemplo, a declaração abaixo cria uma tabela de onda com 10 harmônicos com amplitudes decrescentes, utilizando a GEN 10:

```
f1  0    1024  10  1  .7  .5  .4  .35  .3  .28  .23  .2  .19
```

Deve-se ter cuidado, no entanto, ao se usar tabelas com muitos harmônicos, para que a frequência destes não exceda a sr/2 Hz, causando aliasing[3]. Para isso a fundamental de um som não pode exceder a sr/(2*último harmônico). No caso acima, com sr=44100, e o último harmônico =10, a fundamental não poderá exceder 44100/ 2*10 = 2205 Hz.

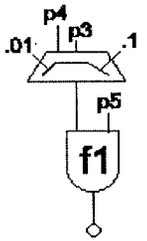
4.3. Geradores de envelope

Outras UG's que são básicas aos algoritmos de síntese são aquelas que geram algum tipo de função de tempo, usadas para fazer algum parâmetro, como amplitude ou freqüência, variar no tempo. No exemplo que mostramos acima na secção 2.2, foi utilizada uma UG chamada *line*, que faz uma interpolação linear entre dois valores em um certo espaço de tempo. Existem, além dessas UG's, outras mais desenvolvidas, chamadas *geradores de envelope*, que geram uma função de tempo com três ou mais segmentos. A mais simples delas é chamada de *linen*, e possue três segmentos, ataque, sustentação e queda (decay). Ela é representada pela seguinte figura nos diagramas de fluxo de sinal:



Uma outra UG um pouco maisdesenvolvida é o ADSR, que possue quatro segmentos: ataque, decay, sustain e release. Esta UG é a utilizada pelo programa audioworks para a geração de envelopes. Envelopes podem também ser gerados por osciladores, utilizando-se funções apropriadas nas tabelas e ajustando-se a frequência do oscilador para 1/duração Hz, fazendo o oscilador passar apenas uma vez pela tabela.

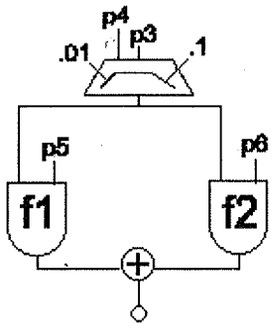
4.4. Conectando UG's




```
instr1
k1      linen p4, p3,.01, .1
aout    oscil k1, p5, 1
        out     aou
```

Como já foi dito, as UG's podem ser conectadas para gerar algoritmos de síntese. A saída de uma UG pode ser usada para controlar algum parâmetro de outra UG. Por exemplo, em caso simples, podemos utilizar um gerador de envelope, do tipo linen para controlar a amplitude de um oscilador, como mostra o diagrama de fluxo acima, traduzido ao lado para a sintaxe csound.

Podemos também utilizar, por exemplo, dois osciladores cujas saídas são mixadas (somadas) na saída do instrumento, com o mesmo gerador de envelope controlando a amplitude de ambos osciladores. Note o uso de dois campos de parâmetros diferentes para as freqüências de cada oscilador, assim como uma tabela de função diferente, garantindo controle individual sobre a freqüência e o timbre de cada oscilador:




```
instr1
        k1      linen   p4, .01, p3, .1
        a1      oscil   k1, p5, 1
        a2      oscil   k1, p6, 2
                out     a1+a2
endin
```

Dessa forma podemos conectar várias UG's para realizar o design instrumental que desejamos. Podemos então estudar certos procedimentos básicos de síntese digital.

4.5. Modulação

Modulação é a alteração da amplitude ou da freqüência de um oscilador de acordo com um outro sinal. Neste arranjo possuimos dois ou mais osciladores, onde o oscilador que está produzindo a modulação é chamado de modulador e aquele que está recebendo a modulação é chamado de portador (carrier). Quando não existe modulação, este gera uma onda contínua chamada de portadora. Quando a modulação é aplicada, a portadora é modificada de alguma forma. Os componentes espectrais de um sinal modulado podem ser classificados em componente da portadora e bandas laterais. A freqüência do componente da portadora é determinada apenas pela freqüência do portador, enquanto a freqüência das bandas laterais é determinada tanto pelo modulador quanto pelo portador.

A modulação de amplitude (AM) pode ser separada em dois tipos principais, a AM propriamente dita e a modulação em anel. Um instrumento que implementa a AM é mostrado abaixo, em sintaxe de csound:

```
instr1
imod  = 1
iamp  = p4
a1      oscil   imod*iamp, p6, 1
a2      oscil   iamp+a1,p5, 1
out     a2
endin
```

Neste caso imod, o indíce de modulação, regula a amplitude do modulador como uma proporção da amplitude do portador. No caso acima, com m = 1, a amplitude do portador é igual à do modulador e diz-se que a modulação é de 100%. Quando as ondas do portador e do modulador são senoidais, o espectro resultante contém energia em três freqüências: a do portador (f_p) e duas bandas laterais (f_p-f_m e f_p+f_m), onde f_m é a freqüência de modulação. A

amplitude da portadora não varia com a modulação, mas as bandas laterais possuem uma amplitude que é m/2 vezes a amplitude da portadora. A freq de modulação também determina a maneira como se percebe o som resultante de AM. Sendo menor que 10 Hz, o ouvinte ouvirá apenas uma ondulação de amplitude no sinal, como um *tremolo*; acima de 10Hz, mas suficientemente baixa para que a portadora e as bandas laterais caiam dentro da mesma banda crítica da audição humana, o ouvido perceberá um som com volume proporcional a amplitude média da onda modulante. Se a f_m exceder metade de uma banda crítica, as bandas laterais vão começar a ser percebidas individualmente.

A modulação em anel acontece quando o sinal do modulador é aplicado diretamente no modulador, sem a soma com um valor representando a amplitude de um sinal sem modulação:

```
instr1
iamp = p4
a1      oscil   iamp, p6, 1
a2      oscil    a1, p5, 1
out     a2
endin
```

Aqui temos apenas as duas bandas laterais, sem a presença da freq da portadora, com amplitudes equivalentes a iamp/2, para portador e modulador com ondas senoidais. O mesmo resultado pode ser obtido pela multiplicação de dois sinais senoidais. Com ondas não senoidais, o espectro resultante será composto pela soma e diferença entre si de todos os parciais das duas ondas, produzindo um espectro muito denso e inarmônico.

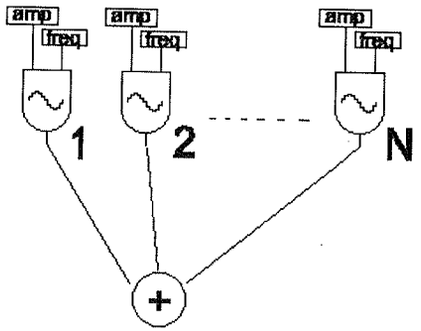
Outro tipo de modulação que é possível de se realizar é a modulação de freqüência (FM), usada para a simulação de vibrato. Neste caso, o portador recebe o sinal modulador em sua entrada de frequência, desviando esta para cima e para baixo de um valor médio:

```
instr1
ilarg = p6
ivel  = p7
ifp   = p5
k1      oscil   ilarg, ivel, 1
a1      oscil   p4, ifp+k1, 2
out     a1
endin
```

No ex. acima, a freq instantânea da portadora vai variar entre " ilarg, a largura do vibrato, a freq média sendo ifp. A largura do vibrato normalmente é apenas uma percentagem pequena de f_p. De forma a ser percebido como vibrato, precisamos manter f_m subsônica (abaixo da gama auditiva humana). Usando f_m acima de 20Hz e uma largura de vibrato (desvio) muito maior, teremos FM como um método de se produzir timbres complexos.

4.6. Síntese aditiva

A síntese aditiva é uma técnica poderosa para a geração de sons complexos, podendo gerar espectros que evoluem no tempo. Ela é baseada na soma dos sinais de saída de um número qualquer de osciladores utilizando ondas senoidais, que são usados para modelar cada harmônico presente em um certo espectro. Cada oscilador tem sua freqüência e amplitude controladas por funções de tempo independentes. A síntese aditiva pode teoricamente gerar qualquer som que possa ser descrito em termos da evolução de seus componentes espectrais no tempo. As funções de tempo para o controle da freq e amplitude podem ser obtidas por análise de sons reais, e neste caso, a síntese é basicamente uma recomposição do espectro analisado. Além disso, em utilizando-se este métode de síntese, pode-se aplicar uma série de modificações interessantes nos espectros dos sons analisados, pode-se interpolar os valores de dois espectros obtendo um terceiro intemediário, etc.. Evidentemente, trata-se de um método computacionalmente muito intensivo, pois requer N osciladores controlados por 2N funções de tempo para um espectro com N harmônicos. Outra dificuldade é a necessidade de se manejar um número muito grande de parâmetros e UG's no design de um instrumento, como se pode observar no digrama de fluxo abaixo:



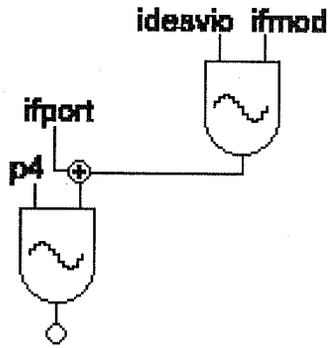
Csound possue uma UG chamada *adsyn* que realiza síntese aditiva a partir de uma som analisado. Neste caso, a UG já incorpora um certo número de osciladores senoidais (até 50), que são controlados por um arquivo de dados de análise, gerado por um programa auxiliar de análise (hetro). Este programa recebe um arquivo de áudio como input e gera um arquivo de dados para ser usado pela UG *adsyn*. Neste caso, trata-se de um meio bem mais conveniente de se empregar processos de síntese aditiva, sem a necessidade de se conectar osciladores individualmente. Além disso, pode-se gerar arquivos de dados para o controle de *adsyn* de outras maneiras, apenas tendo o cuidado de criá-lo dentro do formato esperado. Dessa forma, pode-se criar programas que construam espectros a partir de informações fornecidas pelo usuário.

5. Técnicas de síntese por distorção

Um outro grupo de métodos de síntese é uma opção para a geração de sons complexos ao método aditivo. Estes métodos são chamados de síntese por distorção ou não-linear. Eles são computacionalmente mais eficientes que a ténica aditiva, utilizando um número menor de osciladores para criar um espectro com muito mais componentes que o número de osciladores. As duas técnicas de síntese não-linear mais importantes são modulação de freqüência (FM) e waveshaping.

5.1. FM

A síntese de áudio por meio de modulação de freqüência foi um método desenvolvido pioneiramente por John Chowning na década de 60, aplicando conceitos já desenvolvidos para a transmissão de sinais de rádio por FM. Ela possibilita a criação de espectros complexos que evoluem no tempo, com o uso de apenas dois osciladores senoidais, em um arranjo chamado de FM simples. Com isso a economia em tempo de computação é muito grande com relação à síntese aditiva. Um instrumento para a realização de FM simples pode ser o mesmo utilizado para a produção de vibrato, mostrado na secção 4.5, aplicando-se uma frequência de modulação dentro da gama de audição e um desvio de freqüência maior que o utilizado para vibrato. O instrumento abaixo exemplifica esse arranjo:



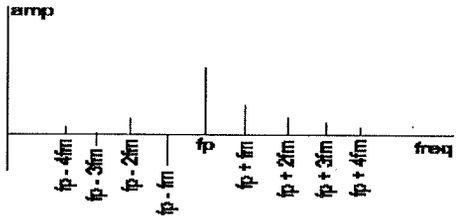

```
instr1
idesvio = p7
ifmod   = p6
ifport  = p5
amod    oscil    idesvio, ifmod, 1
aport   oscil    p4, ifport+amod, 1
        out      aport
endin
```

Como vemos, o modulador recebe um valor chamado *idesvio*, equivalente ao pico de desvio de freqüência, pois a freq do portador (f_p) é desviada para " *idesvio*. O desvio controla a quantidade de modulação inserida no processo. Se o devio é grande é possível que o valor instatâneo da freq do som resultante assuma valores negativos, neste caso o oscilador terá um incremento negativo de fase, fazendo com que esta se mova em direção contrária na tabela. O oscilador apresentado como exemplo na secção 4.1 não é capaz de realizar isso, por isso seu uso para FM é limitado. No entanto, a maioria dos osciladores, em linguagens como csound, podem trabalhar com incremento negativo.

É útil examinar-se o espectro resultante da síntese FM simples, onde os dois osciladores utilizam ondas senoidais. No entanto o espectro resultante pode ser muito rico em harmônicos.

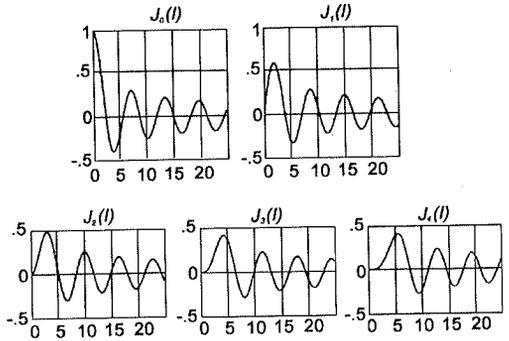
O espectro de FM é composto da freq da portadora (f_p) mais um número de bandas laterais, cada uma delas espaçada a uma distância que equivale à freq de modulação (f_m). As freqs presentes no espectro podem ser carcterizadas pela expressão f_p " kf_m , onde k é um número inteiro que pode assumir qualquer valor maior ou igual a zero. A distribuição de potência entre os parciais depende em parte do *desvio* (d), e quanto maior d, maior a distribuição de potência entre os parciais e maior o número de bandas laterais com amplitudes significantes. A amplitude individual de cada componente espectral depende tanto do desvio quanto da f_m. Usa-se definir um valor chamado de *índice de modulação* (I), que equivale à razão d/f_m, que pode ser interpretado como a quantidade de modulação inserida no sinal. A amplitude de cada banda lateral depende do índice de modulação, segundo as funções de Bessel do primeiro tipo. A amplitude da banda lateral *k* é dada por $J_k(I)$, onde *J* é a função de Bessel, *k* é a ordem da função e o *I*, o índice de modulação. A amplitude da portadora é o caso k = 0. A tabela mostra as bandas laterais da FM simples até k = 4, equivalente ao espectro abaixo(que inclue a comonente f_p):

	inferiores		superiores	
K	Freq	amp	Freq	Amp
1	f_p-f_m	- J_1(I)	F_p+f_m	J_1(I)
2	f_p-2f_m	J_2(I)	F_p+2f_m	J_2(I)
3	f_p-3f_m	- J_3(I)	F_p+3f_m	J_3(I)
4	f_p-4f_m	J_4(I)	F_p+4f_m	J_4(I)



Os gráficos das funções de Bessel de ordens 0 a 4 são mostrados abaixo. Note que para I = 0, todas as funções tem valor zero, exceto a de ordem 0, que controla a componente f_p, pois se não há modulação, toda a potência do sinal reside na freq da portadora, como esperado. Os gráficos também mostram que, dependendo em I, a amplitude do componente pode ser negativa. Amplitudes negativas significam um componente fora-de-fase (com um desvio de fase equivalente aπ rad ou 180 graus), que são representados como no exemplo acima com barras para baixo. Eles vão ter efeitos notáveis somente se outro componente de mesma freq estiver presente, neste caso os componentes serão somados ou subtraídos, de acordo com seu sinal. Os componentes resulatantes de FM também podem ter freqs negativas, e neste caso, também serão interpretados como fora de fase e somados/subtraídos se possuirem o mesmo

valor absoluto de frq de outros parciais. Os gráficos das funções de Bessel indicam também como I precisa ser grande para que as amplitudes das bandas de maior ordem sejam significantes. Em geral, pode-se dizer que as últimas bandas presentes equivalem a k=I+1, com I arredondado para o mais próximo número inteiro.



A relaço entre f_p e f_m pode definir que tipo de componentes espectrais estarão presentes. Se a razão entre as duas freqs puder ser expressa por números inteiros pequenos, um espectro harmônico será resultante. Se a razão for representada por números inteiros maiores ou por números fracionais, o espectro resultante será inarmônico. A fundamental de um espectro harmônico é encontrada achando-se primeiramente a relação f_p / f_m = N / M, onde N e M são números inteiros sem fatores comuns. A fundamental então poderá ser achadá pela relação f_m / M ou f_p / N . Se M = 1, então o espectro poderá possuir todos os harmônicos da fundamental. Se M for um número acima de um, então todo harmônico múltiplo de M estará faltando no espectro. Para M = 1 ou M = 2, os valores absolutos das frequências negativas presentes no espectro coincidirão com as positivas, combinando-se com elas, o que não acontecerá para valores de M maiores que dois. Por exemplo, se f_p = 400Hz e f_m = 200Hz , teremos N = 2 e M = 1, e a fundamental será 200 Hz e teremos todos os harmônicos do espectro. Se f_p = 200Hz e f_m = 400Hz , então N = 1 e M = 2, a fundamental também será 200Hz, mas todo harmônico par estará faltando no espectro.

Para finalizar esta discussão sobre FM, cabe dizer que uma das grandes vantagens deste método é a possibilidade de se gerar, com grande facilidade, espectros que evoluem no tempo. Para realizar isso, tudo o que precisamos fazer é adaptar o design de nosso instrumento FM para que o modulador tenha sua amplitude controlada por uma função de tempo. Podemos usar um gerador de envelope para isso, como no exemplo abaixo:

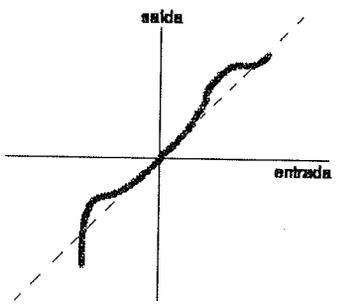
```
instr 1
indice = p7
ifmod = p6
ifport = p5
kdesvio  linen    indice*ifmod, .5, p3, .5
amod     oscil    kdesvio, ifmod, 1
aport    oscil    p4,ifport+amod, 1
```

```
        out     aout
endin
```

Neste exemplo também modificamos a maneira como o desvio máximo vai ser calculado. Utilizamos um valor *indice* multiplicado pela f_m para determinar o valor máximo do envelope, pois assim temos um melhor controle sobre o conteúdo espectral do sinal de saída. O timbre do som será transformado continuamente de uma senóide de freq f_p para o timbre complexo com índice p7, em .5 s, voltando ao som senoidal a .5 s do final. Muitos outros designs podem ser realizados com FM, com possibilidade de realização de diversos timbres, utilizando-se mais de dois osciladores, em diferentes arranjos[4].

5.2. Waveshaping

A outra técnica importante de síntese por distorção é chamada de Waveshaping. Trata-se do uso de uma distorção de amplitude de um sinal como meio de produção de timbres complexos. Aqui utiliza-se um oscilador senoidal cuja saída é conectada a um procesador não-linear, ou waveshaper, que altera a forma da onda passando por ele. Um waveshaper é caracterizado por sua *função de transferência*, que relaciona a amplitude do sinal de saída com o de entrada. Podemos mostrar a função de transferência de uma forma gráfica, como no exemplo abaixo:



A linha pontilhada mostra uma função de transferência linear, onde todos os valores da entrada serão passados à saída sem distorção, f(x)=x. Já a linha mais grossa mostra uma função de transferência que vai alterar nãolinearmente os valores de entrada, causando uma distorçào no sinal. Note-se que essa distorção só ocorrre se o sinal tiver uma amplitude razoável, pois a função de transferência perto da origem é quase igual à curva linear. Com isso a distorção e o espectro resultante varia com a amplitude do sinal de entrada. A maneira mais conveniente de se definir uma função de transferência é algebricamente. Para se evitar aliasing no processo de waveshaping, utiliza-se um polinômio de forma $F(x) = a_0 + a_1 + a_2x^2 + ... + a_nx^n$, para se expressar a função de transferência. Quando se utiliza umaonda senoidal como entrada do processador não-linear, uma função de transferência de ordem N poderá gerar harmônicos somente até o harmônico N.

Um tipo de função de transferência muito usada para waveshaping é o polinômio de Chebyshev de primeiro tipo. Esses polinômios têm a característica de que quando uma onda coseno com a amplitude 1 é aplicada a um waveshaper que utiliza um polinômio de Chebishev de ordem N, a saída é composta apenas pelo harmônico N. Com isso, podemos combinar uma série de polinômios de Chebishev de diferentes ordens em uma função de transferência única para gerar um espectro qualquer desejado quando a amplitude do sinal de entrada for 1.

Para se implementar um instrumento que use waveshaping, precisamos de um oscilador e um processador não linear. O processador não-linear pode ser criado utilizando-se uma UG que leia valores de uma tabela onde esteja armazenada uma função de transferência. Existem GEN's que podem gerar uma função de transferência através do uso de polinônios de Chebishev. Um instrumento que usa waveshaping, juntamente com um exemplo de *score* para utilizá-lo é apresentado abaixo:

```
instr 1
k1      line    1, p3, 512      ; controle de modulação
a1      oscili  k1, p5, 1       ; oscilador senoidal com interpolação
a2      tablei  a1, 2, 0, 512   ; processador não-linear, faz leitura da tabela de acordo
                                ; com os valores de entrada do sinal a1
aout = a2*p4                    ; multiplicação pela amplitude
        outs aout
endin

; score
; co-senóide
f 1   0    1024   9  1  90
; função de transferência:
;          tamanho    GEN  intervalo  amp.   harmônicos:
;                                              DC  1    2   3   4   5   6   7
f2    0    1025       13    1          1      0   10   5   3  .33 2  .5   1

;         dur   amp       freq
i1  0     2     10000     100
e
```

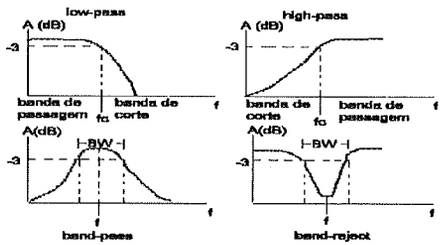
A GEN 13 gera uma função de transferência de acordo com o espectro definido nos parâmetros. Este espectro será gerado quando se utilizar um oscilador senoidal com amplitude amp * tamanho da tabela/2. Neste caso com amp definida na tabela como 1 (no campo p6), o espectro definido será gerado com um sinal senoidal de amplitude 512. Além disso, a tabela têm origem em seu centro (512). Por isso, o último parâmetro para a UG *tablei* é um valor de offset, 512, fazendo com que a leitura seja feita a partir do meio da tabela. Então, o sinal do oscilador, no momento de modulação máxima gerará valores entre 512 e -512, que serão usados pelo processador não-linear (somados a 512) para obter valores da função de transferência armazenados nas respectivas posições da tabela. O sinal será multiplicado pela amplitude geral,

antes da saída do instrumento. A UG line é usada para fazer com que a amplitude do oscilador cresca de 1 a 512, modificando a forma da onda do instrumento de uma senóide para uma onda com os harmônicos definidos na tabela 2. É importante o uso de interpolação pelas UG's para que se evite mais distorções no processo de waveshaping.

6. Síntese subtrativa.

O modelo da síntese subtrativa utiliza uma fonte de excitação e um filtro. Dois tipos gerais de fontes são normalmente usados, uma fonte de ondas periódicas contendo harmônicos com amplitudes similares, chamada de *pulso*, e gerdores de ruído (UG's chamadas *rand, randi ou randh*). A fonte periódica pode ser implementada de duas formas, utilizando-se um oscilador que leia uma tabela onde foi armazenado um ciclo de onda do tipo *pulso* com um certo número de harmônicos, ou pelo uso de uma UG geralmente chamada *buzz*, que gera pulsos com um certo número de harmônicos diretamente. Filtros mudam a característica de um som complexo moldando o seu espectro. Podem ser caracterizados pelo que se chama de *resposta de frequência*, que pode ser separada em *resposta de amplitude* e *resposta de fase*. A resposta de amplitude mostra o quanto uma determinada frequência será atenuada ou amplificada. A reposta de fase indica o quanto a fase de um componente individual é modificada.

Os filtros podem ser de quatro tipos gerais, de acordo com a forma de sua resposta de amplitude: low-pass[5] (LP), high-pass (HP), band-pass (BP) e band-reject (BR):



O primeiro atenua significantemente freqs acima de uma freqüência de corte (f_c), e deixa passar com pouca atenuação freqs abaixo dela. Já o HP faz o contrário, atenuando as frqs abaixo da freq de corte. A f_c , que separa a banda de corte da banda de passagem, geralmente é definida como aquela freq onde a potência transmitida pelo filtro cai pela metade (-3dB)[6] da potência máxima trasmitida na banda de passagem. O filtro BP atenua freqs acima e abaixo de duas freqs. de corte. Ele geralmente é caracterizado por uma freq central f e uma largura de banda (bandwidth) BW. A freq central é média aritmética das frqs de corte superior e inferior e a BW é a distância em Hz entre elas. Também pode ser utilizada uma medida equivalente ao fato de qualidade Q = f / BW, que determina o quão estreito é um filtro. Por último, o filtro BR funciona de forma inversa ao BP, atenuando freqs em uma banda, com freq central e BW.

Um tipo geral de filtro encontrado em linguagens como csound é chamado de *reson*, trantando-se de um filtro BP de segunda-ordem, com controles de BW e f. A ordem de um filtro

geralmente implica na eficiência em que o filtro corta as freqs da banda de corte. Filtros de maior ordem tendem a cortar melhor as freqs indesejadas. Pode se definir a qualidade de um filtro também pela atenuação / oitava, onde um filtro com N dB/oitava de atenuação exibe uma atenuação de -N dB cada vez que a freq dobra na banda de corte. O filtro *reson* possue, em geral, uma atenuação de 6 dB/oitava. O instrumento abaixo mostra o uso desse filtro com uma fonte periódica:

```
instr1
iffreq = p6
ibw  = p7
a1      buzz     p4,  p5,  25, 1 ; fonte periódica com 25 harmônicos
aout    reson    a1, iffreq, ibw  ; filtro
        out      aout
endin
```

Filtros com resposta mais complexa podem ser criados utilizando-se mais de um *reson* conectados de duas formas: em paralelo ou em série. Em conexões paralelas, a saída da fonte é aplicada simultâneamente aos filtros. Este arranjo faz com que as repostas de freqüências de todos os filtros sejam somadas, criando uma série de picos no espectro, equivalentes às diferentes frequências dos filtros. Esses picos podem ser usados, por exemplo, para a simulação de formantes da voz. Em conexões em série, a saída de um filtro é conectada na entrada de outro. Neste caso, a resposta de amplitude do filtro resultante é calculada pela multiplicação das repostas individuais dos filtros usados. Aqui a presença de um filtro com uma certa banda de passagem não garante que o conjunto de filtros passará um sinal com energia significante naquela região, pois algum outro filtro pode estar atenuando severamente aquela região. Comum neste arranjo é a utilização de uma função de balanço, que ajuste o ganho de saída do filtro com relação à amplitude do sinal de entrada, por causa da grande perda de energia do sinal quando da passagem pelo filtro. O instrumento abaixo mostra este tipo de conexão:

```
instr1
iffreq1 = p6
ibw1  = p7
iffreq2 = p8
ibw2  = p9
iffreq3 = p10
ibw3  = p11
a1      buzz     p4,  p5,  25, 1       ;
afil1   reson    a1, iffreq1, ibw1     ; filtro 1
afil2   reson    afil1, iffreq2, ibw2  ; filtro 2
afil3   reson    afil2, iffreq3, ibw3  ; filtro 3
aout    balance  afil3, a1             ; balance o sinal do filtro 3 afil3 com o sinal de entrada
a1
        out      aout
endin
```

7. Conclusão e sugestões de leitura

Este tutorial deverá servir com uma introdução geral aos métodos básicos de síntese. Não foram cobertos aqui alguns métodos desenvolvidos mais recentemente como síntese de formantes (FOF), granular e modelamento físico. No entanto, existe uma literatura muito boa a respeito. Para uma introdução à FOF e aspectos de síntese granular desse método, recomendo o apêndice 3 do manual do Csound, "A FOF Synthesis Tutorial", de Michael Clarke. Para uma introdução ao modelamento físico, especialmente ao algoritmo Karplus-Strong, veja o capítulo 6 do livro de Ken Steiglitz indicado na bibliografia. Para o estudo de designs instumentais interessantes, veja os instrumentos desenvolvidos por J C Risset para o seu catálogo de sons sintetizados por computador (também na bibliografia).

Notas

1. Lembre-se que estamos trabalhando com um sinal digital, que é uma seqüência de valores amostrados sr vezes por segundo representando os valores de um sinal contínuo nos instantes de tempo da amostragem. Para maiores detalhes sobre a teoria envolvida no processo de amostragem, veja (Dodge & Jerse, 1985), cap .1, (Steiglitz, 1995), cap.3 ou (Moore, 1990), cap. 2.
2. Utilizo aqui a palavra inglesa sample para designar o valor assumido pelo sinal digital em um período de amostragem. A tradução poderia ser amostra, mas para clareza prefiro utilizar o termo original.
3. Aliasing, ou foldover, acontece se existem freqüências no som digital que excedem sr/2 Hz, a chamada freqüência Nyquist, inserindo algum tipo de distorção indesejada no sinal. Veja a bibliografia indicada na nota 1.
4. Outros designs muito interessantes podem ser encontrados em (Dodge & Jerse, 1985), cap.4. pp. 113 - 124.
5. Em português, costuma-se traduzir por: passa-baixa, passa-alta, passa-banda e rejeita-banda. Utilizo os termos em inglês para maior clareza e por achar que o termo passa uma tradução ruim do termo pass (embora não no momento consiga sugerir algo melhor).
6. A potência deum som é a energia irradiada em todas as direções por unidade de tempo. A medida em dBs da potência é expressa como 10log P / Pref , onde Pref é uma potência de referência (10-12 Watts). A cada -3 dB, a potência cai pela metade de seu valor.

Bibliografia

Chowning, J. (1977)."The synthesis of Complex Audio Spectra by Means of Frequency Modulation". *Computer Music Journal* 1 (2). Cambridge, M A: MIT Press.

Dodge, C. & Jerse, T (1985). *Computer Music: synthesis, composition and performance.* New York: Schirmer Books.

Lazzarini, V E P (1998). "A Proposed Design for an Audio Processing System ". *Organised Sound* 3 (1). Cambridge: Cambridge University Press,.

Manning, P. (1993). *Electronic and Computer Music.* Oxford: Oxford University Press.

Mathews, M. .(1969) *The Technology of Computer Music.* Cambridge, MA: MIT press.

Moore, F. (1990). *Elements of Computer Music.* Englewood Cliffs, NJ: Prentice Hall.

Risset, J C (1969). *Introductory Catalogue of Computer-Synthesized Sounds.* Murray Hill, NJ: Bell Telephone Labs..

Steiglitz, K. (1995). *A Digital Signal Processing Primer.* Menlo Park, Ca: Addison-Wesley Publ. Co.

Vercoe, B. (1992). *Csound, A Manual for the Audio Processing System.* Cambridge, MA: MIT.

Índice Remissivo de Autores

SBC

Sociedade Brasileira de Computação
Avenida Venceslau Brás, 71 – fundos, casa 27
22290-140 Rio de Janeiro, RJ, Brasil
Fone: (021) 541 8313
Fax: (021) 541 5342
e-mail: sbc@sbc.org.br
URL: http://www.sbc.org.br