TEXTES DES COMMUNICATIONS

Les Actes

des

8e Journées d'Informatique Musicale,

Bourges 7 – 9 Juin 2001

SOMMAIRE


Jim
Bourges 2001
7.8.9 Juin

Les Journées d'Informatique Musicale, Bourges 2001

Les JIM réunissent chaque année des chercheurs en Informatique Musicale et différents acteurs de la vie musicale utilisant l'informatique comme moyen d'expression ou comme aide à la composition.

Cette année, les JIM sont organisées par l' IMEB (Institut International de Musique Electroacoustique de Bourges) en collaboration avec l'ENSI (Ecole Nationale Supérieure d'Ingénieur) de Bourges.

Chaque année les JIM développe particulièrement un thème d'actualité lors d'une session spéciale. Cette année le thème est : *Les Interfaces Gestuelles*.
Dans ce cadre, quatre structures travaillant autour de cette problématique sont invitées à présenter leurs projets : l 'ACROE, la Kitchen, L'IRCAM, le STEIM.

Comité de lecture

Daniel Arfib (CNRS, Marseille)
Gérard Assayag (IRCAM, Paris)
Peter Castine (ICMC, Berlin)
Marc Chemillier (Université de Caen)
Myriam Desainte-Catherine (Université Bordeaux 1)
Christian Doncarli (Ecole Centrale de Nantes)
Emmanuel Favreau (INA-GRM, Paris)
François Giraudon (IMEB, ENSAD)
Jean-Luc Hanus (ENSI, Bourges)
Henkjan Honing (NICI, University of Nijmegen / Music Department, University of Amsterdam)
Vincent Idasiak (ENSI, Bourges)
Héléne Laurent (ENSI, Bourges)
Danny Openheim (IBM, USA)
Yann Orlarey (GRAME, Lyon)
François Pachet (Sony-CSL, Paris)
Miller Puckette (UCSD)
Jean-Michel Raczinski (CEMAMu, Paris)
Sylviane Sapir (Centro Tempo Real, Italie)
Marie-Hélène Serra (IRCAM, Paris)
Todor Todoroff (ARTeM, Mons, Belgique)
Horacio Vaggione (Université Paris 8)
Geraint Wiggins (City University, London, UK)

Comité d'organisation

- Partie Scientifique:
François Giraudon (IMEB, ENSAD), Héléne Laurent (ENSI, Bourges)
- Partie Musicale:
Françoise Barrierre (IMEB, Bourges), Christian Clozier (IMEB, Bourges)

Prix SFIM

Le prix d'une valeur de 5000 Fr. sera decerné au meilleur papier présenté pendant la conférence, l'évaluation se fera a la fois sur la qualité scientifique/technique du papier et sur la qualité de la présentation orale.
Le jury sera constitué du comité de lecture.
Le prix sera decerné a la fin de la conférence.

Editions précédentes

1994 : Bordeaux (LaBRI)
1995 : Paris (LIP6)
1996 : Tatihou (GREYC, Université de Caen)
1997 : Lyon (Grame)
1998 : la Londe les Maures (LMA, Marseille)
1999 : Issy-les-Moulineaux (CEMAMu)
2000 : Bordeaux (SCRIME)

Partenaires

Les JIM sont pilotées par l'ADERIM : Daniel Arfib, Marc Chemillier, Myriam Desainte-Catherine, Yann Orlarey, François Pachet, Jean-Michel Raczinski, Marie-Hélène Serra.

Les JIM sont également soutenues par :

la DMDTS (Direction de la Musique, de la Danse, du Théatre et des Spectacles) ;
la SFIM (Société Française d'Informatique Musicale) ;
L'IMEB (Institut International de Musique Electroacoustique de Bourges)
L'ENSI-BOURGES (Ecole National Supérieur d'Ingénieur de Bourges)

Geste – Modèle physique – Composition musicale

Claude Cadoz

ACROE – ICA
INPG, UJF, Ministère de la Culture
46, Av. Félix Viallet
38000 Grenoble
Tél : 04 76 57 46 61
Fax : 04 76 57 48 89
Mél : Claude.Cadoz@imag.fr

Le geste et la parole, plus généralement les comportements corporels et les émissions vocales sont les deux modalités par lesquelles, au-delà de ses nécessités purement vitales, l'homme est en relation avec son environnement. Le geste et la parole ont comme point commun d'être tous deux des moyens d'expression, mais à la différence de la parole, le geste permet aussi d'agir directement sur le monde physique et de le transformer matériellement. C'est probablement cette ambivalence qui fait en premier lieu du geste un sujet aussi riche et complexe. Mais une autre raison fondamentale le met aujourd'hui au centre des débats : c'est la forte perturbation que son statut a subi à l'arrivée des technologies de l'information et de la communication et le besoin d'en restaurer, dans leur contexte, toutes les dimensions. En effet, tandis que la voix, le son, l'image ont été rapidement convertibles en *signaux* et par la même sont devenus propres à la capture, l'enregistrement, la transmission, l'analyse et le traitement, le geste a vu son champ d'exercice se restreindre comme une peau de chagrin dans les relations avec ou à l'aide des nouvelles technologies. Celles-ci n'en permettaient en effet, jusqu'à récemment, qu'une prise en compte très minimale à travers des dispositifs rudimentaires comme les touches des claviers numériques, réduisant les actes à des sélections et des actions en tout ou rien.
Les questions relatives au geste sont alors bien de deux ordres : celles qui se rapportent à la compréhension de sa nature et de sa diversité, et celles qui se rapportent aux technologies et aux conditions qui permettent de les servir pleinement l'une et l'autre.
En nous plaçant dans la perspective de la création musicale qui recourt, totalement ou partiellement, aux nouvelles technologies, nous proposons ici un parcours schématique pour montrer un lien du geste naturel à la création musicale à l'aide des nouvelles technologies. Ce parcours emprunte un chemin spécifique, déterminé par l'approche même de la création artistique à laquelle nous nous sommes attachés dans nos travaux : celle de l'utilisation de l'ordinateur comme moyen de simulation de l'univers physique. En fait, nous abordons la simulation non seulement comme un paradigme pour la création de situations interactives et la production de phénomènes sensibles, mais également comme un mode d'éclairage et de compréhension général et unificateur.

Les questions que nous allons soulever adressent successivement les natures du geste, les moyens de leur prise en considération dans les relations avec ou à travers les nouvelles technologies, leurs relations avec les processus numériques de création des phénomènes sonores et des structures musicales. Nous tenterons enfin de mettre l'interaction gestuelle en perspective dans le cadre d'une utilisation générale de l'informatique pour la création musicale.

Les natures du geste

Nous reprenons ici en résumé une série de dichotomies simples mais fondamentales qui nous ont permis par ailleurs[1] de proposer un cadre typologique à l'approche du geste.

Le geste en général

Il est tout d'abord apparu nécessaire de distinguer clairement, dans les diverses invocations du geste, celles qui s'y réfèrent de manière métaphorique de celles qui s'attachent à sa réalité matérielle. Certaines dynamiques au sein d'un énoncé musical peuvent présenter des formes temporelles, des allures qui renvoient par évocation au comportement gestuel. Il faut bien noter que cette évocation peut se manifester en dehors de toute réalité effective d'un geste de production ; il suffit pour cela que des variations perceptibles de la séquence sonore présentent des similitudes d'évolutions temporelles avec celles d'un geste possible. Cela indique que notre système cognitif dispose de référents relatifs à ces dynamiques et que ces derniers peuvent jouer un certain rôle dans la structuration de nos représentations mentales. Même si le qualificatif de "gestuel" peut apporter une forme de lecture pertinente de ces dynamiques, il est clair que c'est un usage métaphorique et que l'on ne peut l'aborder avec la même méthodologie que le geste réel.

Une seconde distinction, importante dans l'observation de l'exercice du geste en amont des nouvelles technologies, s'appuie sur le geste en soi, c'est-à-dire où seul le corps est matériellement en cause, par opposition au geste qui s'effectue dans le cadre d'une interaction avec un objet physique. Le premier, dans la mesure où ses manifestations à l'égard de l'environnement ne peuvent être reçues que visuellement, ne peut être que de nature *sémiotique* (selon la terminologie que nous avons proposée [ibid.]), c'est-à-dire qu'il ne peut servir qu'à faire connaître, émettre de l'information. En musique, le geste du chef d'orchestre entre typiquement dans cette catégorie[2]. Les comportements gestuels relatifs à la locomotion doivent être exclus de cette catégorie dans la mesure où le corps interagit avec un objet physique (certes particulier) : le sol avec ses reliefs. Dès lors qu'un objet physique intervient, deux circonstances nouvelles apparaissent : (1) le système physique en jeu est l'ensemble constitué du corps et de l'objet ; ses propriétés et donc ses états et ses évolutions sont différents de celles du corps seul, (2) selon la nature de l'objet, le comportement gestuel peut servir différents objectifs. Il est alors pratique de distinguer deux catégories très différentes selon que l'objet en question interagit matériellement avec l'environnement physique et permet alors de le modifier ou de le transformer (comme une pelle, une pioche ou un marteau) ou qu'il est spécialement conditionné pour produire ou relayer des phénomènes sensibles. On pourra parler d'outils dans le premier cas, d'instruments dans le second. La catégorie qui

---

[1] CADOZ C., "Continuum énergétique du geste au son : simulation multisensorielle d'objets physiques". in "les interfaces pour la création musicale". – Dir. H. Vinet, F. Delalande. HERMES Editeur. 1999.
CADOZ C., "Musique, gestes, technologies". in "Les nouveaux gestes de la musique". sous la direction d'Hugues Genevois et Raphaël de Vivo. 1999.
CADOZ C., WANDERLEY M., "Gesture and Music". in Trends in Gestural Control of Music. IRCAM Editeur. 2000. avec CDROM.

[2] À noter que lorsque le chef d'orchestre a une baguette, il y a bien une interaction avec un objet physique. Le statut de cette situation pourra être clairement défini avec des critères plus fins introduits plus loin.

retiendra notre attention ici est naturellement la seconde mais on peut signaler au passage que ces deux catégories ont en commun le fait qu'une relation énergétique (*ergotique*) s'établit entre le corps et l'objet : indépendamment de la finalité matérielle ou communicationnelle, une dépense effective d'énergie en dehors du corps a lieu, qui met en mouvement ou déforme l'objet physique. Ce dernier peut alors transmettre cette énergie à un tiers (par exemple l'environnement aérien dans lequel se propagera une onde acoustique) ou la restituer partiellement au manipulateur qui, de ce fait dans le même temps reçoit une forme d'information à travers ses capteurs tactiles et proprio-kinesthésiques. Le canal gestuel est dans ce cas un moyen de connaissance (fonction *épistémique*) qui s'ajoute aux autres sens. Notons que le geste instrumental a ceci de particulier qu'il met en jeu de manière indissociable les trois fonctions, *sémiotique*, *ergotique* et *épistémique* que nous avons introduites au fil de ce paragraphe. C'est là sans doute une de ses propriétés les plus importantes. Deux remarques s'imposent alors à ce niveau :

- Le lien entre la fonction *sémiotique* et la fonction *ergotique* peut être très fort : certaines figures dynamiques expressives ne peuvent s'obtenir qu'à la condition que le geste soit pleinement physique (donc *ergotique*). L'aspect *ergotique* est donc partie prenante à part entière du processus communicationnel et il est illusoire de penser que toutes les formes d'information gestuelle peuvent être obtenues à main nue puisque la physique du système dans le cas de la situation instrumentale fait bien intervenir au moins deux composants : le corps humain et l'instrument, et les propriétés de l'ensemble ne peuvent s'obtenir et se décrire qu'en considérant l'ensemble.
- Le lien entre la fonction *sémiotique* et la fonction *épistémique* est également très fort puisque, là aussi, il s'agit d'un système, d'une inter-relation : la conduite des mouvements expressifs se fait, pour certains gestes, dans la logique d'une boucle intime où chaque phénomène de perception tactile et kinesthésique pendant l'action est susceptible de modifier dynamiquement la suite du geste.

Dans la catégorie des instruments, une dernière dichotomie est utile : celle qui permet de mettre d'un côté les instruments qui, du geste humain à la perception, s'appuient sur une continuité énergétique et ne font appel (ce qui est un corollaire) à aucune source d'énergie extérieure à l'homme, et de l'autre les dispositifs qui interviennent pour moduler un flux énergétique (par exemple lumineux) qui s'établit entre une source extérieure et nos sens (par exemple la vision). Le geste instrumental musical traditionnel (c'est-à-dire avant l'électronique et mis à part quelques cas singuliers comme l'orgue) se place dans le premier cas. Le geste du marionnettiste entre dans le second cas puisque les mouvements de la marionnette ne sont visibles que du fait de son interaction avec le faisceau d'un projecteur. Le geste d'écriture (musicale comme littéraire) ou le geste graphique entrent également dans cette seconde catégorie : le crayon ou le pinceau sont des instruments ;, mais le dispositif se complète par une fonction nouvelle que l'on peut appeler d'une manière générale la *gravure* qui est essentiellement la transformation d'une forme temporelle en une forme spatiale. Cette fonction essentielle est la base de tous les processus de mémorisation des phénomènes temporels. Elle se généralise lorsque la trace n'est plus nécessairement directement visible ou lisible sans un intermédiaire, comme dans la gravure magnétique ou "micro-optique" des CD.

En ce qui concerne la musique, ce premier tronçon du parcours nous permet de distinguer au moins quatre types de situations, et donc quatre fonctionnalités différentes du geste : le geste au sens de la métaphore, le geste au sens du chef d'orchestre, le geste dans l'écriture et le geste instrumental. Avant d'aborder le problème du geste dans le cadre des nouvelles technologies, il faut accorder quelque attention particulière au geste instrumental musical.

Le geste instrumental musical

La fonction physique que doit remplir l'instrument de musique est précise : elle est quantitative et qualitative. Qualitative, elle est de permettre la transformation de phénomènes gestuels en phénomènes acoustiques. Quantitative, elle est de permettre, à travers cette transformation, l'émission d'une certaine énergie à destination des auditeurs à partir d'une énergie développée par l'instrumentiste. La seconde fonction ne peut être remplie qu'à la condition que la relation entre l'instrumentiste (qui est la seule source d'énergie) et l'instrument soit une interaction physique. Dès lors, toute action de l'instrumentiste sur l'instrument est indissociable d'une réaction du second sur le premier et l'ensemble instrumentiste-instrument forme en soi un système. Les phénomènes d'interaction dépendent des propriétés de l'un et de l'autre et c'est par la même, en particulier, que l'instrumentiste, à travers ces différents phénomènes, reçoit par ses sens tactiles et proprio-kinesthésiques une certaine information sur les propriétés et les comportements intimes de l'instrument.

Selon le point de vue quantitatif, l'instrument est donc une chaîne énergétique et la première forme du geste instrumental est déterminée par l'objectif de fournir efficacement à l'instrument une énergie qu'il pourra convertir en énergie acoustique. Nous avons nommé cette contribution gestuelle "*geste d'excitation*" [ibid.].

La chaîne matérielle (chaîne instrumentale élémentaire) assurant la transmission énergétique fait appel à des composants matériels qui présentent, par nécessité et par utilité une certaine permanence. En conséquence de quoi, les solutions adoptées pour assurer cette transmission détermineront autant de catégories fondamentales d'instruments. Mais pour un même principe physique de transmission, de multiples variations sont possibles et certaines d'entre elles sont même envisageables à partir d'un même dispositif par modulation ou modification de quelques propriétés simples de certains composants (longueur d'une corde ou d'une colonne d'air par exemple). Les actions gestuelles peuvent être impliquées dans cette tâche de modification qui aura pour effet de moduler le phénomène de génération sonore pendant le jeu instrumental même. Cette seconde composante du comportement gestuel, qui ne s'inscrit pas de la même façon dans le processus énergétique mérite d'être identifiée pour elle-même. Nous l'avons nommée "*geste de modification*".

Enfin, plusieurs dispositifs différents ou correspondant à des variantes physiques d'un même principe peuvent être impliqués simultanément, dans un instrument complexe) pour remplir en séquence ou en collaboration la fonction principale de production sonore. L'instrumentiste s'implique alors gestuellement d'une troisième façon, dont la physique est distincte des deux précédentes. Nous appelons "*geste de sélection*" ce qui correspond à ce choix en ligne, pendant le jeu même, de la ou des chaînes instrumentales. Pendant la sélection (comme dans le cas du jeu pianistique par exemple) l'interaction matérielle avec l'objet peut s'interrompre. Dans le cas nominal, le geste de sélection est donc à main nue et n'entre pas dans le cadre *ergotique*.

Ce rapide parcours montre que la catégorie même du geste instrumental peut être en soi multiforme puisque, construite autour de cette condition énergétique centrale, elle met en jeu les fonctions gestuelles de manières différentes et combinées : la fonction *ergotique* est déterminante dans la composante "*excitation*", elle y est évidemment indissociable des fonctions *épistémique* et *sémiotique*. Dans la composante "*modification*", la fonction *ergotique* peut être présente (comme lorsqu'il s'agit de tendre la peau d'une timbale à l'aide d'une pédale) ou quasiment pas (comme lorsqu'il s'agit de modifier la longueur d'une corde

dans le violon). Pour la composante "*sélection*", il n'existe pas, hormis dans les automates musicaux, de dispositifs chargés de gérer ou d'assister cette sélection. La composante "*sélection*" se caractérise alors par le fait que ce sont pratiquement exclusivement les contraintes physiques et physiologiques humaines qui président à leur conception. En revanche, la fonction *épistémique* du canal gestuel peut y être mise fortement à contribution, lorsqu'il s'agit par exemple de recalibrer à chaque instant le positionnement de la main sur un clavier en s'aidant du contact avec les touches enfoncées ou les touches noires voisines.

Ces quelques considérations de base permettent d'amorcer une méthodologie générale pour parcourir l'espace gestuel et identifier ses différentes natures. On peut constater au passage que le développement de cette analyse se fait en parallèle avec une incursion de plus en plus intime dans la structure même de l'instrument. C'est dire qu'en fait, l'analyse du geste instrumental et l'analyse de l'objet instrument sont indissociables et qu'il serait vain, puisqu'il s'agit fondamentalement d'une interaction, d'envisager l'une sans l'autre.

Un parcours complet de toutes les lignées, espèces et individus instrumentaux qui ont résisté à la sélection humaine serait fort long. Aussi, on se contentera ici d'un seul niveau complémentaire dans l'arborescence : celui des formes d'excitation, pour montrer au passage que si les catégorisations procèdent indubitablement de choix culturels complexes, elles résultent également de contraintes propres aux règles de l'univers dans lequel nous vivons.

C'est le cas en effet pour la catégorisation des instruments et des gestes d'excitation correspondant en ces deux grandes classes que sont les instruments à excitation instantanée (percussion, cordes frappées, pincées, etc.) et les instruments à entretien continu (à frottement d'archer, à vent, etc.). Il n'existe en effet, "dans la nature", que deux façons de communiquer de l'énergie à une structure vibrante à partir d'une structure non vibrante (comme le corps humain), celle où le transfert se fait quasiment instantanément, séparant le geste et le son par une frontière événementielle ponctuelle (fin de l'un, début de l'autre) et celle où le transfert se fait en continu, moyennant une disposition physique très particulière (comme on peut l'analyser entre le crin collophané et la corde) et où geste et son se développent dans le même temps. Les gestes d'excitation correspondant sont évidemment de deux natures complètement différentes, difficiles à ramener l'une à l'autre. Ce qui permet de dénoncer au passage l'ineptie des instruments électroniques qui offrent à jouer des sons de violon en enfonçant les touches d'un clavier.

La création sonore à l'aide des nouvelles technologies

Dans le cadre de la création sonore à l'aide des nouvelles technologies, le geste instrumental ne répond plus à aucune nécessité matérielle. En effet, le propre des systèmes électroniques quels qu'ils soient est d'introduire, par le principe de l'amplification, une coupure fonctionnelle fondamentale dans les circuits énergétiques. Le transistor (et avant lui la triode sont des relais : ils mettent en jeu deux circuits énergétiques indépendants, dont l'un (le circuit de commande) à faible énergie, *module* les conditions de l'autre (le circuit actif) à énergie élevée, à la manière d'un robinet dont on ajuste l'ouverture pour contrôler le débit d'eau. Les quantités d'énergie dans l'un et l'autre circuit peuvent être d'échelles très différentes, en particulier l'énergie dans le circuit de commande peut être négligeable devant celle mise en jeu dans le circuit actif. Par ailleurs, sauf à le provoquer expressément par un dispositif de "*feedback*", il n'y a pas d'échange d'énergie entre les deux circuits. Enfin, si le circuit de commande peut déterminer le flux énergétique du circuit actif, l'inverse n'est pas vrai : la relation est unilatérale.

Le principe du relais, qui peut prendre des formes matérielles très variées est, par son exploitation systématique et généralisée dans les "nouvelles" technologies, l'un des quelques principes technologiques de base qui permettent de les caractériser par rapport aux "anciennes", même s'il est déjà présent depuis l'Antiquité dans des machines comme le premier clavier d'orgue du mécanicien d'Alexandrie Xtésibios en 270 avant J.C. Les notions de signal et d'information elles-mêmes lui doivent l'existence. C'est en effet à partir du moment où autre chose que de l'énergie ou de la matière peut être véhiculé entre deux lieux que l'on parle alors d'information, portée par un signal. Ce sont ces conditions matérielles qui déterminent également les aspects relationnels : à la différence d'une grandeur énergétique ou matérielle, une entité informationnelle ne se véhicule que dans un sens. Dans la même logique, les entités informationnelles peuvent être "diffusées", c'est-à-dire distribuées sans se diviser à plusieurs destinataires (ce qui n'est évidemment pas le cas pour la matière et pour l'énergie). Enfin, les conditions énergétiques à l'émission et à la réception peuvent être totalement indépendantes : l'énergie nécessaire à l'émission peut être faible et les énergies mises en jeu au récepteur très importantes.

Il est clair que dans ce contexte, la première transformation qu'opèrent les nouvelles technologies dans le domaine de la création sonore portent, de manière profonde, sur les conditions matérielles de la production des phénomènes sensibles à partir des actes physiques humains (gestuels). En conséquence, la chaîne de production dans le cadre des nouvelles technologies ne peut plus être analysée selon les mêmes critères et les systèmes électroniques et informatiques dédiés à la création sonore ne peuvent être définis comme procédant simplement d'une "nouvelle lutherie".

En restant au niveau de la production du matériau, par opposition à celui de la création et de la maîtrise de la structure musicale (compositionnelle), l'évolution des approches marque schématiquement trois grandes étapes : une première approche que l'on pourrait qualifier de "phénoménologique", une seconde "fonctionnelle" et une troisième "structurelle".

La première, dont le représentant historique est la démarche de la musique concrète, se caractérise en fait par l'appropriation du signal (sonore) en tant qu'objet et par le développement de toutes les techniques de son traitement et de ses transformations (segmentation, transformations temporelles, fréquentielles, morphologiques, chaînage, superposition). Ces techniques sont le substrat matériel de la construction de "langages", ou au mois de processus de création musicaux spécifiques. Elles se prolongent dans le cadre des technologies numériques et trouvent une forme synthétique et générale dans toutes les approches fondées sur le traitement du signal.

La seconde, qui a vu le jour en particulier avec la synthèse sonore par blocs fonctionnels représentée par le "paradigme Music V" de Max Mathews va au-delà de la surface (phénoménologique) du signal. Elle met en œuvre, en amont, des processus fonctionnels dont l'agencement en structures hiérarchisées (fondées sur la notion de commande de blocs fonctionnels par la sortie d'autres blocs fonctionnels de même nature) engendre le signal avec toutes ses propriétés. Cette approche peut d'ailleurs s'arrimer à la précédente dans la mesure où les signaux engendrés peuvent être également traités. Le substrat des langages et de l'expressivité sonores est dans ce cas déterminé par la base fonctionnelle des modules du système. Dans le cas de Music V est des systèmes qui en dérivent et abritent les différentes méthodes de synthèse développées depuis les années 60 et 70 (synthèse additive, soustractive, FM, FOF, granulaire, etc.), appelés généralement aujourd'hui systèmes de synthèse en modèle

de signal, ces bases relèvent toutes d'un même angle de vue : le phénomène sonore est une entité en soi, décomposable en constituants élémentaires caractérisés par des paramètres (qu'il s'agisse des sinusoïdes ou des grains de Gabor) et tout le problème de la génération sonore est ramené aux différentes manières d'engendrer ces éléments ou des ensembles structurés de ces éléments, et de contrôler leurs paramètres ou des ensembles structurés de leurs paramètres.

La troisième approche, historiquement la dernière, pose le problème d'une façon différente. Représentée par les courants relevant du "modèle physique", elle déplace l'objet d'étude en amont du phénomène, au niveau de sa cause. Elle s'approprie l'objet plutôt que le signal. Alors que les deux approches précédentes partent du signal respectivement pour le traiter ou pour l'engendrer, celle-ci part de l'objet et le simule. Il serait exagéré de caractériser aussi clairement toutes les approches qui se réclament du modèle physique car nombre d'entre elles n'invoquent que très marginalement le paradigme de la simulation et restent très largement dans un environnement général d'exploitation basiquement conditionné par les méthodes en modèle de signal. Mais toutes les approches "modèle physique" en revanche ont comme trait commun de se fonder, en allant plus ou moins en profondeur, sur une analyse structurelle des causes physiques sonores naturelles et une construction des simulacres selon une structure qui tend à suivre à l'identique la structure de l'objet de référence. Ainsi, par exemple, pour réaliser la synthèse d'un son de guitare, on mettra en jeu dans la simulation des composants algorithmiques correspondant respectivement aux cordes, au chevalet, à la caisse, etc. Dans ce troisième cas, la référence aux objets et phénomènes réels intervient comme élément structurant du vocabulaire sonore.
L'approche modèle physique, dans la mesure où elle conduit de toute manière à la génération d'un son numérique se raccorde évidemment aux méthodes de synthèse du son et de traitement du signal.

L'histoire des technologies récentes pour la musique présente, singulièrement, un cheminement inverse de celui de l'évolution des instruments eux-mêmes : on part en effet de la destination (le son) pour remonter à son origine (l'instrument). Ceci démontre d'une autre manière que les nouvelles technologies ne peuvent être l'occasion de nouvelles lutheries, qu'elles se placent dans une position non pas de prolongement ou d'extrapolation mais de "réflexion" (au sens du miroir). Bien sûr, le miroir est actif et c'est là sans doute ce que l'ordinateur instaure, sans équivalence antérieure : la possibilité de créer des "images" des représentations reflétant pour parties les propriétés du monde réel, mais capables aussi d'y intégrer des propriétés nouvelles. Resterait ensuite à voir comment ces images peuvent éventuellement susciter la création de leurs originaux réels.

Le geste et la machine

La "machine" d'aujourd'hui est informatique ou en tout cas au moins "informatisée", ce qui signifie que les phénomènes physiques dont elle est le siège sont, indépendamment des fonctions qu'ils remplissent, de nature électronique et numérique. L'homme, quant à lui, est toujours biologique, c'est-à-dire que les phénomènes physiques par lesquels il est en relation avec ce qui l'entoure (par exemple une machine) sont des ondes acoustiques, des ondes lumineuses, ... pour ses récepteurs et des mouvements mécaniques pour ses comportements et perceptions gestuels. Interagir gestuellement avec une machine suppose donc expressément une fonction qui n'avait pas lieu d'être dans les technologies antérieures, puisqu'il s'agit de transformer des phénomènes mécaniques (ou pneumatiques lorsqu'il s'agit du souffle) en phénomènes électroniques numériques. Cette fonction de "transduction" (littéralement transformer / conduire) s'est posée pour tous les domaines d'application de l'électronique. Par

exemple, il a fallu créer des transducteurs acoustico-électriques pour transformer les vibrations acoustiques en signaux électriques ou des transducteurs électro-acoustiques (les... hauts parleurs) pour la transformation inverse. La première condition pour exercer un geste à l'adresse d'une machine est donc que celle-ci dispose de transducteurs gestuels.

Les technologies des transducteurs gestuels sont variées, mais néanmoins déterminées par un petit nombre de principes. Un premier critère va les séparer en deux catégories principales selon la nature du phénomène pris en considération pour représenter le geste. En effet, le geste est un phénomène par nature mécanique, mais il peut aussi se manifester, au moins partiellement, sous forme visible. En conséquence de quoi l'on peut décider de capter des mouvements mécaniques ou des variations lumineuses (c'est ce que fait l'homme dans le sens de la "perception gestuelle"). Dans le premier cas, un intermédiaire matériel est nécessaire, avec lequel doit s'établir un contact physique direct ; la relation gestuelle pourra être, au moins localement, instrumentale. Dans le second cas, aucune interaction physique ne s'impose. Il s'appuie typiquement sur la capture visuelle, par exemple par camera, et sur une ensemble de technologies d'extraction de données comme les contours de la main, des membres, du corps de l'opérateur et, à partir de celles-ci, de variables représentatives du comportement corporel[3]. Loin de correspondre à une catégorie secondaire, il ne permet cependant d'exploiter que la fonction *sémiotique* du geste. Le geste concerné est en conséquence exclusivement non-instrumental.

Les technologies de la transduction mécanico-électrique sont, quant à elles multiples. Elles sont toutes basées sur la production ou la modification d'un phénomène électrique en conséquence au déplacement ou à la déformation d'un objet (que celui-ci soit solide ou fluide). Le cœur de la fonction de transduction repose sur quelques principes, en nombre limités : variation de conduction électrique (d'effet résistif), variation d'effet inductif (déplacement d'un corps magnétique à proximité ou à l'intérieur d'une bobine ou d'antennes), variation d'effet capacitif, effet piezo-électrique, effet de champ, pour les principaux. On peut y ajouter les variations de transmission d'ondes ultrasonores, lumineuses, ou électromagnétiques en général.

Indépendamment de la technologie de transduction proprement dite, un autre critère intervient à ce niveau, qui partage à son tour les dispositifs en deux catégories selon que l'objet soumis à déplacement ou déformation est attaché au corps et se présente en quelque sorte comme une prothèse ou un prolongement de celui-ci, ou qu'il est face à l'opérateur qui peut choisir d'entrer en contact avec lui ou non. Dans le premier cas, on parle de techniques immersives, dans le second de techniques de vis-à-vis. La relation avec l'objet et ses significations possibles ne sont pas les mêmes. Dans l'une, la technologie prolonge l'homme, dans l'autre elle prolonge son environnement.

---

[3] Un dispositif célèbre auquel il faut rendre hommage car il fut le tout premier, en 1920, et celui qu'inventa Theremin. Exploitant un effet capacitif entre la main et deux antennes sphériques, il permettait le contrôle sans contact de la fréquence et de l'amplitude d'oscillateurs électroniques. Contrôlant des grandeurs relatives à la « substance » sonore, il n'est cependant pas instrumental, du fait de l'absence de contact. Il est le premier exemple historique introduisant une rupture entre la nature du geste et la nature des phénomènes contrôlés.

Dans la première catégorie, les divers dispositifs développés et exploités aujourd'hui se rangent à leur tour dans quelques catégories de base[4] représentées par le gant et les combinaisons de données (*dataglove, datasuit*), les exosquelettes de la main, du bras, du corps, etc. Il faut mettre dans ces catégories le cas des objets "embarqués" par la (ou les) main(s) ou le corps et qui portent soit des sources (lumineuses, infra lumineuses, infra ou ultrasonores, électromagnétiques) émettant vers des capteurs situés dans l'enceinte du jeu corporel, soit, à l'inverse, des récepteurs de ces mêmes catégories de phénomènes. Dans tous les cas, les informations captées et transmises sont des représentations des comportements corporels intrinsèques et non d'une interaction physique réelle, même si parfois la lourdeur des systèmes "embarqués" peut modifier la dynamique et les trajectoires des mouvements. Il s'agit donc, malgré la présence et le contact avec un objet physique, d'une relation purement *sémiotique* et donc non-instrumentale.

Dans la seconde catégorie, la plus simple et la plus ancienne est l'interrupteur, utilisé dans le manipulateur de télégraphe aussi bien que dans les claviers alphanumériques d'aujourd'hui. L'image du geste, en tout cas dans sa fonction *d'excitation* telle que nous l'avons définie, est réduite à sa plus simple expression : instant de contact et quantification en tout ou rien. Mais une série d'interrupteurs est en revanche bien adaptée à la capture de la composante de *sélection*. C'est cette dimension exclusive que les claviers d'instruments électroniques ont exploité essentiellement pendant longtemps. Le déplacement d'un curseur sur une piste conductrice, que celle-ci soit rectiligne ou circulaire, donne une image moins squelettique du geste puisqu'elle peut porter des variations "analogiques". Les évolutions gestuelles sont toutefois réduites à une seule variable. L'"habillage" physique du curseur (bouton à déplacement rectiligne ou rotatif, touche à grand enfoncement, bras articulé, etc.) permet d'exploiter ce principe dans une gamme de situations variées pouvant s'approcher de situations de référence instrumentales naturelles. Avec un habillage minimal, ce type de transducteur est un candidat raisonnable au support de la fonction de *modification* (dans sa modalité analogique, par opposition à "discrète" [ibid.]). Mis en batterie, il peut de plus jouer pour la "sélection-de-modification". Il faut en revanche que l'habillage en question soit un peu élaboré, en incluant par exemple une mécanique spécifique (dont la plus simple est le ressort de rappel), pour se rapprocher de quelques conditions naturelles élémentaires du geste *d'excitation*. Il faut bien noter que ces conditions seront alors figées pour un dispositif donné.

C'est dans cette seconde catégorie, avec une disposition nouvelle, qu'il faut situer les transducteurs gestuels rétroactifs (TGR). Ils peuvent exploiter l'une quelconque des technologies précédentes pour capter les phénomènes d'interaction opérateur-objet physique, mais ils portent une fonction symétrique supplémentaire : des transducteurs électromécaniques (en d'autres termes des moteurs électriques) qui peuvent alors assurer un retour de la machine vers les capteurs tactiles et proprio kinesthésiques de l'opérateur. L'ensemble des caractéristiques d'un TGR est alors déterminé par celles de ses trois composants fonctionnels : les capteurs, les effecteurs et son "habillage". Les premiers et les seconds sont déterminés quantitativement par les caractéristiques de précision, dynamique et bande passante. Elles doivent être adaptées à celles des variables gestuelles en cause : les déplacements gestuels peuvent être significatifs dans une plage importante (du micromètre au mètre), les efforts peuvent aller de quelques centièmes à quelques centaines de newtons. La

---

[4] On pardonnera cette énumération purement catégorielle et trop rapide pour évoquer nommément les dispositifs ingénieux et fort pertinents de nombreux collègues dont nous apprécions tout particulièrement le travail.

bande passante des phénomènes gestuels (en force et en déplacement) peut aller de 0 jusqu'à plusieurs KHz (à noter qu'il a fallu un certain nombre d'années pour prendre conscience de cette nécessité et que nombre de systèmes à retour d'effort ont pâti longtemps de la croyance que quelques dizaines de Hz suffisaient). L'"habillage" enfin est un caractéristique importante. Il s'agit schématiquement des conditions géométriques (morphologie et trajectoires) et dynamiques selon lesquelles l'objet physique réel (et incontournable) se présentera à la saisie et à la manipulation de l'opérateur. Enfin, un TGR peut être constitué d'un ensemble de chaînes capteurs-effecteurs conférant au tout un certain nombre de degrés de liberté. Rappelons que la main, en tant que système mécanique, comporte à elle seule 23 degrés de liberté. À noter également que l'habillage peut être envisagé comme s'appliquant non pas à chaque DDL terme à terme, mais à des groupements structurés de DDL, pour permettre par exemple des manipulations de type joystick, pince, déplacement d'un solide tridimensionnel, etc.

La technologie des TGR pose un problème particulier, sans équivalent pour les autres transducteurs (visuels ou acoustiques) : du fait de la nécessité d'un intermédiaire matériel réel, il ne peut exister de dispositions universelle permettant la prise en charge de toutes les catégories d'interactions physiques possibles. Les TGR sont alors nécessairement catégorisés en fonction de leur application. Néanmoins, il est possible, comme le propose la technologie que nous avons développé à l'ACROE[5] de mettre en jeu une forme de généricité fondée sur la modularité des trois composants de base (en particulier l'habillage).

L'introduction de la rétroaction permet de prendre en charge la fonction *ergotique* du geste et, par conséquent de la composante *excitation* du geste instrumental d'une façon non plus figée, mais programmable, puisque les comportements mécaniques dynamiques de l'objet manipulé pourront être contrôlés par la machine, dans une boucle d'interaction effective. Cette boucle pourra associer, par des algorithmes appropriés, les données d'entrées gestuelles à la production de comportements mécaniques s'adressant aux capteurs tactiles et proprio kinesthésiques de l'opérateur.

Systèmes gestuels et synthèse

Nous disposons maintenant, avec les processus de synthèse et les transducteurs gestuels, de tous les ingrédients permettant d'établir la relation du geste à l'oreille en utilisant un ordinateur. Il existe naturellement un grand nombre de façons d'associer les premiers et les seconds, dans la mesure où la seule condition pour qu'il se passe quelque chose est que les variables issues des systèmes gestuels et celles que les dispositifs de synthèse requièrent se correspondent, au moins quantitativement. Toutefois, notons que pour la prise en compte de la rétroaction, donc pour soutenir le geste *d'excitation* en particulier (assurer la fonction *ergorique* en général), les dispositifs de synthèse doivent également produire les signaux de "sortie gestuelle" (les signaux de commande des moteurs de rétroaction).

---

[5] CADOZ (C), LUCIANI (A), FLORENS (JL), 1984, « Responsive Input Devices and Sound Synthesis by Simulation of Instrumental Mechanisms : The Cordis System », Computer Music Journal, 8, N°3, pp. 60-73. M.I.T. Press, Cambridge Mass.
CADOZ (C), LISOWSKI (L), and FLORENS (JL), (90) - "A Modular Feedback Keyboard Design", Computer Music Journal, Vol. 14, No.2, Summer 1990.

La relation entre les données gestuelles et les données de synthèse peut être posée en termes banals de "contrôle" et de "*mapping*". Contrôle au sens où l'on considère que le son est le produit d'un algorithme quelconque et que les paramètres d'entrée de celui-ci sont des variables de statut non nécessairement déterminé, ou répondant aux paramètres acoustiques classiques. "*Mapping*" dans le sens où l'on considère les aspects relatifs au geste et relatifs au son séparément et indépendamment dans un premier temps, pour établir leurs correspondances dans un second temps. Les critères de correspondance peuvent alors évidemment être larges et ne s'appuyer que sur les caractéristiques des signaux analysées de manière neutre. Cette approche laisse le champ ouvert à toutes les utilisations possibles, mais au-delà d'un tel objectif de généralité, nous préférons développer une approche guidée par la recherche de ce qui peut rendre le geste et la production sonore cohérents entre eux.

Systèmes gestuels et simulation des objets physiques

La synthèse sonore par modèles physiques particulaires permet d'aborder dans un même principe, et dès le niveau élémentaire, la création des phénomènes sonores et des phénomènes mécaniques correspondant aux comportements des objets au niveau gestuel. La particule inertielle élémentaire (définie comme une masse ponctuelle dans un espace physique) et l'interaction viscoélastique (linéaire ou non-linéaire) sont en effet deux composants de base qui permettent, par leur combinaison en réseau de forme quelconque, de simuler des mécanismes aux comportement vibratoires et d'établir la correspondance physique requise entre des variables (forces ou déplacements) représentant les grandeurs relatives aux actions gestuelles, et les variables (déplacements ou forces) représentant les grandeurs relatives aux réactions de l'objet. Des variables gestuelles aux variables sonores (et éventuellement d'ailleurs aux variables visuelles, dans le cadre d'une multisensorialité généralisée), il n'y a pas de changement de système de représentation, seulement des changements d'échelles et de propriétés, les unes et les autres pouvant être abordées à l'aide d'un même langage général et d'un même nombre restreint de concepts.

L'ensemble de la chaîne instrumentale peut alors être mis en représentation, simulé sous la forme d'un instrument virtuel. Cet instrument virtuel, mécanique dans sa conception, pourra alors être "joué" à l'aide d'un ensemble de transducteurs gestuels, parmis lesquels on disposera des TGR, de façon à réaliser une relation instrumentale. La relation gestuelle avec l'instrument virtuel pourra ainsi être réellement instrumentale.

Nos travaux concernant la musique, à l'ACROE, s'attachent à cette situation, en mettant en œuvre, à l'issue des travaux de recherche correspondant, un langage de modélisation et de simulation des objets physiques par réseaux particulaires, le langage CORDIS [ibid.], des transducteurs gestuels rétroactifs, en particulier à partir de la technologie du Clavier Rétroactif Modulaire [ibid.], un environnement de création, l'outil GENESIS.
À l'aide de CORDIS et GENESIS[6], nous avons réalisé la modélisation da la plupart des catégories de dispositifs instrumentaux, en nous fondant sur une décomposition de la chaîne instrumentale en ses composants de base : excitateur, structure vibrante, chevalets-tables, environnement de rayonnement, etc. Pour chacun de ces composants, les catégories de base

[6] CADOZ (C) "Musical creation with Multisensorial Interactive simulation of Physical Objects" - SCI 2001 – Orlando – à paraître – 2001.

ont été modélisées : structures vibrantes simples (cordes, colonnes d'air, plaques, membranes, volumes, ensembles de particules. etc.), modes d'excitation par percussion, pincement, entretien continu (archet, anches, etc.), caisses et tables de résonance, milieux propagatifs simples... Les résultats se concrétisent par un instrumentarium large permettant de créer la plupart des phénomènes sonores naturels et instrumentaux, ainsi que de créer des objets de nature physique mais introuvables dans la nature. Il est possible également, par cette approche, d'envisager la création sonore en partant de l'idée du son voulu ou en référence, a priori, et en déterminant, grâce à une méthodologie précise quel processus mécanique (réaliste ou non) permet de le produire. L'environnement permet naturellement de créer des ensembles instrumentaux qui, lorsque les machines sont assez puissantes, les TGR et les instrumentistes respectivement en nombres suffisants, peuvent être joués en exécution d'une partition ou en improvisation comme s'il s'agissait d'instruments réels.

En attendant que les machines atteignent (ce qui n'est pas une utopie) les puissances requises pour que le temps-réel soit applicable à une échelle significative, nous terminons ce parcours avec la mise en perspective du processus de création musicale tel qu'il peut être d'ores et déjà pratiqué aujourd'hui avec les moyens actuels.

Geste – modèle physique particulaire – Composition musicale

Simuler "aussi" l'instrumentiste

L'instrumentiste est un être humain, intelligent et sensible. Mais, sans que ceci n'enlève rien à cela, c'est aussi un système physique. À ce titre, la partie qui ne concerne que ce dernier aspect peut être (moyennant quelques simplifications) modélisée et simulée, avec par exemple un langage tel que CORDIS et un environnement tel que GENESIS.

Les figures gestuelles sont d'une variété infinie : gestes des doigts qui pianotent, qui grattent ou pincent, gestes de la main qui attrape, tire, pousse, caresse, frappe, pose, gestes des bras qui bercent, moulinent, lancent, gestes du corps qui trépigne, sursaute, balance, etc. Mais quelques figures de base, à condition qu'elles soient pratiquées par les bons éléments ou les bons ensembles, permettent de les engendrer plus ou moins toutes. Ce sont les figures respectivement du *déclenchement* de *l'accompagnement* et de la *balistique*.

Le déclenchement résulte de la décision, à un moment donné, d'entrer en action avec une intensité variable, depuis la simple pichenette jusqu'au frappé violent. Deux grandeurs physiques peuvent en représenter les caractéristiques : la vitesse et l'inertie. Alors, une masse ponctuelle d'une inertie donnée, lancée avec une certain vitesse initiale, peuvent jouer le rôle, au moins pendant un moment, de la partie d'un instrumentiste qui voudrait pincer, percuter ou pousser quelque chose. Il suffit alors, pour simuler l'instrumentiste à ce moment et pour cette action, de simuler une telle masse et de la faire interagir avec les objets d'une scène instrumentale qui pourraient se prêter à ce jeu.

L'accompagnement et la balistique sont deux formes semblables par un point, elles durent et la nature de leur évolution est leur substance – et différentes par un autre, chaque instant du premier est voulu et contrôlé, tous les instants de la seconde sont déterminés une fois l'action commencée. Chez un pianiste, lorsqu'il a lancé son corps, séant en bascule sur le siège, buste en balance au-dessus, épaules en roulement, bras et avant-bras en fléau dans leur prolongement pour exécuter un trait de Chopin, l'impulsion, dès qu'elle est lancée, du bas du

dos, se propage comme une onde jusqu'au crépitement des doigts sur l'ivoire. Cette colonne de propagation peut être modélisée comme un assemblage de composants inertiels en interaction viscoélastique (avec toutes nos excuses auprès des pianistes). Plus qu'une simple corde, dont il faudra toutefois que les fréquences propres soient de l'ordre des fréquences gestuelles (basses, de 0 à 20 Hz), une sorte de chaîne se dédoublant puis se décuplant, possède la balistique d'un tel geste.

On voudra moduler cette balistique et garder le contrôle des doigts, un par un, il nous faudra recourir, pour les combiner astucieusement, à nos déclencheurs précédents. On voudra, encore, différemment, frotter par un geste souple et modulé, il nous suffira de "lisser" quelques impulsions judicieusement envoyées par les mêmes déclenchements avec un peu de coefficient de frottement visqueux au bon endroit.

L'instrumentiste réel, loin d'être détrôné, trouvera là une nouvelle place. Au jeu réel sur des instruments réels, il peut ajouter le jeu réel sur des instruments virtuels, et aussi jouer d'une nouvelle façon en duo, trio, ... en "écoutant" ses partenaires (virtuels) non plus seulement avec son ouïe, mais avec son corps et ses gestes... (tout en respectant toutes les règles de bienséance en public).

Composer

Le compositeur n'est pas détrôné non plus, puisqu'il peut toujours, si c'est son désir, composer (poser ensemble) les notes sur une partition et les faire jouer, mais il peut de plus composer des instruments, composer des instrumentistes, et enfin, grâce à cette objectivation nouvelle, celle de "l'objet gestuel" capté, enregistré, représenté, composer le geste.

VERS UN VETEMENT INTERACTIF

Thierry CODUYS, Gilles DUBOST

La KITCHEN
5 rue Laugier 75017 Paris
(Thierry.Coduys@la-kitchen.fr)
(Gilles.Dubost@la-kitchen.fr)

Résumé :
Dans cet article, nous décrivons dans une première partie un projet de recherche sur un "vêtement interactif" permettant de suivre le mouvement du corps et d'analyser les gestes. Dans une seconde partie nous décrivons un prototype de "corset interactif" que nous avons développé et décrivons les champs d'investigation qu'il suscite. Ce corset fera l'objet d'une démonstration à l'occasion des JIM 2001.

I. INTRODUCTION

La création artistique s'est toujours intéressée à l'étude du geste comme moyen d'expression extrêmement riche (danse, musique théâtre...) et à son interprétation.
L'outil informatique et le traitement du signal permettent aujourd'hui, par le biais de capteurs, de suivre avec précision le geste, de l'analyser et de l'exploiter.
La plupart des systèmes de mesure (ou de captation) existants se basent sur deux approches différentes : l'une exploite des capteurs de proximité, qui sont fixés sur la personne en mouvement (Dataglove, système Biomuse, etc.), l'autre exploite des mesures sans contacts à partir de systèmes entourant la personne (sonars, systèmes électromagnétiques, systèmes infrarouges, vidéo temps réel associée à un traitement de l'image, etc.). Dans le premier cas, on peut obtenir des données fort précises et subtiles sur les gestes effectués d'un individu donné, mais les technologies employées jusqu'à aujourd'hui ont souvent perturbé le geste ou gêné la mesure par leur simple présence (poids des appareillages, câbles, etc.).
Dans le second cas, on réalise surtout une analyse comportementale globale et non individuelle. En outre, ce type de mesure confine la personne en mouvement dans un espace délimité (celui où la mesure est effective), contextuel.
Les progrès constants réalisés dans le domaine de la miniaturisation des capteurs, du traitement numérique du signal et de l'image, du transport sans fil des données permettent aujourd'hui à de nombreux laboratoires et sociétés d'améliorer sensiblement l'exploitation du geste selon les deux approches.

Dans le cadre d'une collaboration avec le couturier GUCCI, nous nous sommes intéressés à la fibre optique, et à la fibre en général. Le textile, qui comme de nombreux secteurs prend en compte les avancées technologiques, gagne lui-même des parts de marchés sur d'autres matériaux dans des secteurs comme l'aéronautique, l'automobile, la bagagerie ou encore le sport. Il s'agit dorénavant pour les industriels de valoriser le textile dans tous les aspects de la vie quotidienne :

développer les textiles à vocations industrielles mais également ceux orientés directement vers les consommateurs (habillement, ameublement, loisirs, spectacles...)
C'est pourquoi un marché de plus en plus important s'intéresse à l'intégration de la technologie dans le textile et en particulier dans le vêtement : on assiste à une émergence significative des vêtements communicants, en particulier dans les domaines de l'art et de la médecine. Chercheurs et entreprises s'associent pour créer des prototypes dans le but de les commercialiser prochainement. Des marques très réputées telles que Levi's, Adidas, Siemens ou encore Courrèges travaillent actuellement dans cette direction. Des couturiers, des petites sociétés, des organismes se sont également mises à la tâche (Olivier Lapidus, Wearlap Technology, I-Wear, France Télécom R&D, Centre de recherches textiles néo-zélandais Wronz, Peratech Ltd, etc.).
L'intégration de la technologie dans le textile n'a pas uniquement pour but l'esthétique et le confort ; elle peut agir également sur le corps humain, l'hygiène, l'adaptation climatique et la sécurité. On peut imaginer que des interfaces hommes/machines seront intégrées dans toutes les surfaces textiles imaginables, tel le projet anglais "softswitching" ("contact souple").
En 1990, le Ministère de la Santé a créé le laboratoire Dermscan, chargé entre autres de recherches sur les nouvelles fonctionnalités des textiles innovants. De nombreux centres de recherches comme le CNRS, ainsi que certains laboratoires développent largement les possibilités offertes par le textile, ou effectuent des recherches comportementales. Des études sociologiques et technologiques sur l'évolution du textile au fil des siècles donnent lieu à des conférences et forums.

Comme le montrent ces différents exemples, le marché des technologies innovantes liées au textile est actuellement en pleine expansion.
A La Kitchen, nous avons décidé de développer la première approche décrite précédemment, à savoir l'analyse du geste par une détection de proximité et individualisée, parce qu'elle nous paraît être la plus riche et la plus pertinente. Les textiles technologiques permettent aujourd'hui de palier à de nombreux problèmes et offrent même de nouveaux horizons. Nous avons donc élaboré la définition d'un "vêtement interactif", qui a fait l'objet d'un brevet déposé en 1999, dont nous présentons ci-après le principe en détail.
A titre d'exemple, nous présenterons aux JIM 2001 un prototype de corset "interactif", que nous décrivons ici dans une seconde partie.

II. LE VETEMENT INTERACTIF

Le principe de ce vêtement consiste à intégrer dans du textile des capteurs, de l'électronique et des systèmes intelligents, à tisser du fil électrique, de la fibre optique, et de nouveaux matériaux de façon à créer un tissu permettant de détecter et d'analyser le mouvement corporel.
Cette technologie permet de suivre et d'interpréter les mouvements corporels et de générer les actions en conséquence - déclencher en temps réel des actions ou des processus sur ou dans le vêtement lui-même, ainsi que pour communiquer avec des entités externes (déclenchement et contrôle à distance). En outre, elle permettra également d'établir une communication sensorielle entre le vêtement et son porteur (son, vibrations, chaleur...).
Selon le type d'application (aide aux handicapés, sport, ergonomie du travail, création artistique...), le vêtement interactif générique pourra être programmé et configuré de façon adéquate.

Ce vêtement est destiné à un large public, dans des domaines d'applications variés tels que l'aide aux handicapés, aux personnes âgées, la médecine du sport, l'ergonomie du travail, ou plus particulièrement la création artistique.

Ce projet propose une nouvelle définition du vêtement : à ses deux principales fonctions (protéger le corps, habiller le corps) – que l'on modernise (nouveau matériaux, fibres éclairantes, etc.) – , il en apporte une troisième : exploiter le corps et son mouvement.
Des analyses corporelles existent depuis longtemps, mais sont pour la plupart cantonnées en laboratoires, et elles ne permettent pas une interprétation en temps réel et *in situ*. Le vêtement interactif, par l'intégration complète en son sein des moyens d'analyses, permet enfin de suivre le mouvement du corps et de l'exploiter sans gêner aucunement le geste.
Les applications sont multiples, concernent tous les individus sans exception, et pourront à terme influer sur le comportement et le mode de vie de chacun d'entre nous.

Le concept de vêtement interactif rassemble plusieurs technologies innovantes :
- il se base sur toutes les recherches et les nouveaux procédés concernant le "textile technique" : en particulier la fibre, brique élémentaire de la conception textile, revêt aujourd'hui de multiples formes et fonctions : fibre électrique, optique, métallique, naturelle, biotechnologique, à mémoire de forme, etc.
- il intègre toutes les technologies de captation, dont les plus récentes : capteur d'accélération, de rotation, de champ magnétique, de flexion, de torsion, etc.
- il fait appel à la miniaturisation sans cesse croissante de l'électronique et son intégration dans de nouveaux matériaux ainsi que son implantation dans le milieu biologique.
- il utilise des technologies très élaborées concernant sa fiabilité, sa résistance à l'eau, au lavage, à la température, etc., ainsi qu'à son caractère inoffensif vis à vis du corps humain.
- son utilisation n'est pas cantonnée au domaine du gadget ou du simple contrôle. Elle est le fruit d'une réflexion et d'une recherche approfondies, basées d'une part sur une interprétation sophistiquée du geste et du mouvement (analyse sociologique, définition d'une sémantique, d'une grammaire de gestes, etc.) et d'autre part sur les technologies de pointe permettant l'analyse et l'exploitation intelligente des données issues des capteurs (microprocesseurs, DSP, traitement numérique du signal, intelligence artificielle, réseaux de neurones, etc.).

Etudes techniques préalables déjà réalisées

Dans le cadre de l'activité de La Kitchen, les études techniques concernant directement ou indirectement la définition du vêtement interactif ont été réalisées sur une période de recherches effectuées entre 1999 et 2001.
Ces recherches ont eu lieu notamment avec le concours du CNRS (laboratoire IRCOM de Limoges) et d'entreprises partenaires (Ruband Gallant et Dubar Warneton, Atelier 33), et sont également le fruit problématiques soulevées lors de travaux réalisés pour de grandes sociétés : GUCCI, RATP, France Telecom.

Les différentes études réalisées ont porté sur le :

- développement de technologies de capteurs, leur conditionnement et leurs applications (capteurs de flexion, pression, température, photoélectrique, accéléromètre, piezzo, magnétique, sonar, etc.)
- développement d'électroniques de traitements, travail sur la miniaturisation (CMS) et les techniques de circuits imprimés souples (projet d' "écharpe alphanumérique" avec France Telecom)
- développement de technologies de transfert de données sans fil (liaison HF).
- travail sur les sources lumineuses (leds, lasers, etc.), les moyens de créer, véhiculer et diffuser la lumière.
- travail sur les sources d'énergie (accumulateurs, cellules photovoltaïques, etc.)
- travail sur le son (sources diffusantes, sources sonores miniaturisées, directivité, bande passante, etc.) en collaboration avec la société Atelier 33 (traitement acoustique)
- travail sur le traitement numérique des données.
- travail pour la société italienne GUCCI sur la faisabilité de l'intégration industrielle de technologies de capteurs et d'animations lumineuses dans des sacs en cuir (réalisation de 3 prototypes).
- travail, en collaboration avec des tisserands, sur la conception de tissus technologiques intégrant la fibre optique éclairante, des fils électriques (fil de LITZ), des fibres métalliques (cages de Faraday locales anti-ondes GSM).
- travail, avec des compositeurs, des artistes et des interprètes, sur les aspects philosophiques et sociologiques du geste, son utilisation par l'outil informatique ou électronique afin de cerner la problématique gestuelle liée au vêtement interactif.

Les applications

L'aide aux handicapés :

Le vêtement interactif peut constituer, dans le domaine de l'aide aux handicapés, un apport aussi considérable que l'invention du velcro. Ce vêtement a l'avantage de n'être ni un lourd appareillage ni une orthèse, et ne distingue pas son porteur d'une personne en bonne santé.

Grâce au vêtement interactif, un moyen de communication entre l'handicapé et son environnement peut être établi :

- dans le cas d'un handicap psychomoteur, le porteur du vêtement aura la possibilité d'agir à distance sur le milieu qui l'entoure (ouvrir une porte, programmer, organiser) par le biais de machines (moteurs ou autres) avec lequel le vêtement communique.
- dans le cas de dysfonctionnement des organes sensitifs, le vêtement interactif peut permettre d'établir un nouveau mode de communication entre la personne handicapée et son environnement. A titre d'exemple, un sourd-muet pourra en fonction d'une gestuelle précise faire énoncer des mots par un haut-parleur intégré au vêtement, et établir ainsi une communication qu'il ne peut aujourd'hui pas avoir avec une personne ne maîtrisant pas le langage des signes. Inversement, un système de reconnaissance de la parole peut informer une personne sourde en émettant dans le vêtement des vibrations acoustiques (un langage vibratoire, composé de différentes fréquences, est imaginable). On imagine sans peine des applications similaires pour les personnes atteintes de cécité.
- Enfin, le vêtement interactif peut intégrer les systèmes de contrôles utilisés dans le cadre d'implants bio-technologiques, et permettre de nouvelles innovations dans ce domaine.

Ainsi la suppression d'activités occasionnée par un handicap peut être en partie supplantée par le port du vêtement interactif. Le vêtement interactif est un produit généraliste, mais il peut également être programmé en fonction des besoins de l'handicapé. Il est donc possible de le fabriquer en masse tout en l'adaptant à chaque individu.
Il concerne bien sûr également les personnes atteintes de maladies moindres (épilepsie, vertiges, etc.), ainsi que les personnes âgées auxquelles il peut rendre des services semblables à ceux décrits précédemment (suppléer à des déficiences physiques), et permettre un système sécuritaire (surveillance, appel, etc.).

L'ergonomie du travail :
L'analyse des mouvements du porteur du vêtement permettra une meilleure compréhension et donc une amélioration de sa posture. Les accidents seront ainsi en partie évités et le bien-être du porteur accru.
De plus, l'efficacité des mouvements du porteur sera facteur d'une plus grande rentabilité dans son travail.

Le sport :
Dans le milieu sportif, le vêtement interactif va permettre, par rapport aux analyses déjà existantes, une analyse du mouvement corporel de proximité indépendamment du lieu d'étude. Ainsi ces analyses pourront être réalisées hors d'un laboratoire et avoir lieu sur un terrain de compétition ou d'entraînement.
Le vêtement interactif peut trouver des applications chez les amateurs comme chez les professionnels : le vêtement peut fournir des données personnelles à un sportif amateur, évaluer ses performances et suivre son évolution. Dans le domaine professionnel, en sport individuel comme en sport collectif, il peut évaluer les performances mais également tendre à leurs améliorations sans recourir à des moyens illicites...
Le vêtement interactif peut également analyser les réactions du corps dans les pratiques de sport à haut risques, tels que la plongée sous-marine ou le saut en parachute, et informer le sportif des données environnementales, renforçant ainsi sa sécurité.
Le vêtement interactif permet une meilleure connaissance du corps humain en situation de stress, de compétition ou simplement de loisir sans occasionner de gêne pour le sportif.

La création artistique :
L'analyse du geste propose un nombre infini d'applications dans les domaines suivants :
- la scénographie
- la chorégraphie
- la musique
- la mode
- la réalité virtuelle

L'intégration de cette analyse dans un vêtement autorise de nouveaux possibles : son action n'est plus nécessairement liée à un lieu (la salle de spectacle par exemple). On se déplacer, agir dans la rue comme ailleurs. En outre, les processus créatifs peuvent être individuels ou collectifs : nous nous dirigeons vers l'ère de l'homme augmenté ou de l'homme réseau...

Un premier exemple de vêtement est donné ci-après avec le "corset interactif".

III. UN CORSET INTERACTIF

Description

Le prototype de corset que nous avons réalisé intègre les principales technologies qui seront employées dans le vêtement interactif.

Ce corset est lumineux : conçu de fibres optiques tissées et traitées (de façon à devenir diffusantes latéralement), il offre une surface éclairante de 20cm x 20 cm. La superposition et l'entrecroisement des fibres permet de donner au corset différentes couleurs.

Deux capteurs extrêmement sensibles se connectent sur le corset. Il peuvent se placer à tout endroit du corps jugé pertinent. Le premier est un capteur magnétique sensible au champ magnétique terrestre et que l'on emploie comme inclinomètre. Le second est un accéléromètre sensible au champ gravitationnel et aux accélérations dynamiques.

Enfin, le corset peut communiquer avec une entité externe (en particulier un ordinateur) par liaison HF, et ce de façon bidirectionnelle.

Champs d'explorations, applications

La lumière

Le traitement de la lumière et son interaction avec le son fait l'objet depuis plus d'un an de recherches à La KITCHEN, qui se sont matérialisées par la création, avec le compositeur André Serre-Milan d'une œuvre intitulée "...Toiles filantes..." exploitant le "Galiléographe", machine lumineuse inventée par Laurent Bolognini et Françoise Henry. Cette œuvre a été présentée à l'ISEA 2001, ainsi qu'au forum de L'IRCAM (octobre 2000).

Dans le cas du "Galiléographe", nous nous sommes intéressés aux possibilités d'écriture d'une sémantique lumineuse et à l'interaction lumière-musique (accords, désaccords, paradoxes, trompe-l'œil, trompe-l'oreilles, etc.).

Dans le cas du corset, nous nous intéressons à des questions similaires : quelle est la nature de l'interaction qui peut exister entre un geste et la lumière qu'il génère (les capteurs influent sur la lumière diffusée) ? Le geste est-il interprété différemment si la lumière diffusée est indépendante du mouvement (le corset reçoit des informations de l'ordinateur, indépendamment des données fournies par les capteurs) ? Quel paramètre lumineux s'associe au geste : la dynamique, le rythme, la couleur ? Comment perçoit-on une personne dont le vêtement change d'apparence en fonction de son mouvement ? Etc.

En outre, la multiplicité de tels corsets dessine de nouvelles perspectives passionnantes : il est possible d'éclairer simultanément, ou selon des procédés divers, plusieurs personnes qui peuvent se mouvoir librement dans l'espace : vers un pilotage à distance d'"hommes luisants"...

Le geste musical

Comme nous l'avons dit précédemment, le suivi du geste par une mesure de proximité nous paraît être la méthode la plus riche et la plus convaincante. Sans parler des capteurs biotechnologiques (de Biomuse aux connections directes sur les synapses), il existe aujourd'hui pléthore de capteurs qui peuvent, isolément ou en combinaison avec d'autres, fournir des informations précises sur le geste : accéléromètre, inclinomètre, microphone, transducteur piezzo-électrique, capteur de flexion, de température, etc.

Dès lors, nous nous intéressons à La Kitchen, à la mise en œuvre de telles technologies, mais également à leur utilisation vis à vis du geste musical.
Ainsi, les deux seuls capteurs que nous associons au corset (inclinomètre et accéléromètre) ouvrent des champs d'exploration passionnants : s'ils fournissent bien sûr des données pertinentes pour le contrôle ou la sélection d'un paramètre musical (paramètre de synthèse, volume, choix d'un son, d'une note, etc.) auquel est associé le geste, leur finesse et leur excellente définition suscitent notamment deux axes de recherche.
Le premier concerne la reconnaissance de dynamiques et de formes : par l'étude simultanée des capteurs et un traitement du signal approprié, il est possible d'identifier et de reconnaître un geste donné. L'utilisateur (le musicien, le danseur, ...) peut alors tenir un discours de gestes, discours musical, lumineux, verbal, etc.
Le second porte ce que nous appelons à La Kitchen "l'avant geste", geste qui précède ou prépare le geste excitateur. En particulier, l'accéléromètre qui permet le suivi de gestes amples, s'avère très pertinent dans le traitement, par exemple, des gestes percussifs. Avec un tel capteur, il devient alors possible de proposer un travail sur la matière sonore (ou autre) qui pourrait précéder la génération du son que le geste est sensé traditionnellement et physiquement provoquer. On peut imaginer également que seul "l'avant geste" ait lieu, et travailler alors l'absence du geste excitateur attendu, et sur les effets d'illusion sonore que cela peut provoquer.
Qu'il soit utilisé uniquement pour l'analyse d'un geste préparatoire ou à des fins créatives, "l'avant geste" nous apparaît comme un champ d'investigation passionnant, et qui suscitera sans doutes d'autres possibles : "l'après geste", "l'entre geste" (ou inter gestes), etc.

Enfin, le corset interactif tel que nous le présentons, permet de travailler sur la globalité des multiples interactions qu'il provoque entre le geste, l'intensité lumineuse, la couleur, et les matériaux (sonores, vidéo, ou autres) auxquels sont associés les capteurs.

IV. QUELQUES REFERENCES

Principales technologies employées :
http://www.atmel.com
http://www.analog.com

Capteurs :
http://www.entran.com/
http://www.frc.ri.cmu.edu/robotics-faq/10.html
http://www.cs.sfu.ca/people/ResearchStaff/amulder/personal/vmi/HMTT.add.html
http://www.iee.lu/fsr1.htm
http://www.imagesco.com/articles/articleindex.html
http://www.msiusa.com/sensors.htm
http://www.imagesco.com/catalog/flex/FlexSensors.html
http://www.stetson.edu/~mdemurga/interactive/links.html
http://www.billbuxton.com/InputSources.html

Logiciel :
http://www.cycling74.com/index.html

Suivi et analyse du mouvement :
http://www.daimi.aau.dk/~diem/digitaldance.html
http://musart.dist.unige.it/
http://metwww.epfl.ch/physilog/Physilog.htm

Vêtement communiquant :
http://www.starlab.org/bits/intell_clothing/
http://www.internetquintessence.net/globalcomposites/news/newsbysector/technology/peratech.html

Etat du contrôle gestuel à l'Ircam

Andrew Gerzso & Marcelo Wanderley
IRCAM
gerzso@ircam.fr

Résumé : Les résultats récents dans le domaine du contrôle gestuel reflètent surtout une diversité d'approches. Le contrôleur de violon SuperPolm permet de piloter la synthèse utilisant un gestuel similaire à celui d'un violoniste. Escher est un système pour la création d'applications de contrôle gestuel implémenté en jMax, un environnement de programmation visuel temps réel.
Par ailleurs, un contrôleur à vent MIDI a servi comme interface pour le pilotage en temps réel d'un modèle physique de la trompette. L'analyse des mouvements des clarinettistes a également été étudie à fin de mieux comprendre les gestes d'un musicien pendant une exécution musicale. Enfin, une interface pour le contrôle de Chant en Max/MSP utilisant une tablette Wacom a été utilisé dans la simulation d'un archet de violoncelle. Nous
présenterons l'état actuel de ces travaux ainsi que les projets de leur utilisation dans les productions futures de l'Ircam, notamment dans le domaine de la danse.

Gestural Control at Ircam.

This report presents an overview of Ircam activities related to gestural control in music over the last five years. We comment on the different activities, from music creation and courses, to developments in hardware and applied research.

1. Historical Perspective

Since its inception, Ircam has been interested on how to control sound synthesis gesturally (Sequential Drum) [MathewsBennet78].
In the eighties, a man-machine interface project was carried out [Battier86] that has given light to developments such as the MIDI flute [Pousset92] and the PACOM [StarkierPrevot86] that have both been extensively used in compositions and musical productions at the institute and abroad.
G. Dubost wrote a dissertation (D.E.A.) on the theme: ``Sensor techno log and their Musical Applications' [Dubost93]. As a result of this work, a sonar type sensor was built, that was used initially by X. Chabot [Chabot93]. Among other researchers/composers, A. Tanaka has also taken part in related projects, such as a children workshop in previous Ircam Open House events (1995).
Interest in historical perspectives has led to consultancy with the Musée de la Musique in Paris [Battier93] and an ongoing collaboration with Electronic Music Foundation (EMF) (http://www.emf.org/institute/index.html).

2. Current Activities

The report focuses on the period form 1996 to the present (February 2001).
During these five years, various activities have been carried out related to gestural control at Ircam. They are summarized below.

- **Publications**

Two publications have been edited at Ircam concerning gestural control and man-machine interfaces:
- Interfaces Homme-Machine et Création Musicale, Hermes Science Publications - 1999, edited by Hugues Vinet and Francois Delalande (GRM) is the revised proceedings of the workshop held at Ircam in December 1998.
Invited speakers have made a detailed review of several fields related to human-computer interaction in music, such as graphical interfaces, musical controllers, interactive systems, etc.
- Trends in Gestural Control of Music, Ircam - Centre Pompidou - 2000, edited by Marcelo M. Wanderley and Marc Battier, is an Electronic Publication (CDROM) containing varions articles and tutorials on gestural control of music. Specifically, it consists of:

a) a round table with leading instrument builders and performers,
b) twenty-four articles, tutorials and case studies on gestural control of music,
c) an extensive bibliography (around 500 entries), and
d) a detailed list of resources. It includes several videos and sound examples illustrating the articles, which are especially formatted for screen reading and printing.

Research and Development Many projects focusing on or related to gestural control are cure-ongles being developed in the Research Department, headed by Hugues Vinet. These projects relate to new instrument design and the control of different synthesis methods, physical and signal models.

- **Alternate Controllers**

P. Pierrot and A. Terrier developed the SuperPolm violin controller for composer/performer S. Goto [Goto99]. Seven sensors integrated in a violin-like controller provide 11 continuous output variables. The ensemble provides an interface that is able to control different synthesis parameters departing from a gesticulation similar to that of a violin player.
A. Terrier, P. Pierrot and X. Rodet collaborated with on the development of JERRY, a four dimensional mouse - a normal computer mouse equipped with two pressure sensors, allowing the simultaneous control of 4 continuous parameters [RodetTerrierPierrot97]. Real-Time Additive Synthesis Control – ESCHER The development of a real-time synthesis system with applications to gestural control, based on j-MAX has been carried out from the end of 1997 by Butch Rovan, Norbert Schnell, Shlomo Dubnov and Marcelo Wanderley. The system is intended to provide a flexible environment for sound synthesis and a basis for experimentation in human-computer interaction in music [RWDD97][WSR98]. It is built as independent modules that may be replaced according to the synthesis method used, and to the type (level) of interaction desired.

- **Haptic devices**

During a sabbatical leave in 97/98, Vincent Hayward (McGill University) worked collaboration with Butch Rovan, what led to the development of a tactile feedback system to be used in conjunction with open-air computer music performance devices [RovanHayward00]. The VR/TX system is proposed as a solution for adding tactile feedback to open-air controllers.

- Real-time control of Physical Models

Research by Christophe Vergez and Xavier Rodet focuses on the control of a physical model of a trumpet in real-time [RV96] [VR01]. The control interface was developed using a MIDI wind controller for the information concerning mouth's pressure and piston position. The resulting interaction between the player and the model closely simulates the one experienced by a performer and his instrument. Stefania Serafin has used the WACOM tablet (Cf. below) to control a physical model of a violin [SDWR99].

- Modelling Performer Gestures

Still regarding research topics, M. Wanderley and Ph. Depalle develop research on gesture capture [DTW97], modeling and application to the control of sound synthesis by signal models [WDW99]. The analysis of clarinetists movements has been carried out in [Wanderley00] and [Wanderley01] in order to understand the basis of performance gestures. Wacom Tablet In collaboration with CNMAT, a research project on the use of the WACOM tablet as an instrument controller has been carried out in 1999 and 2000, funded by the France-Berkeley Fund. During this period, a MSc. thesis by Jean-Philippe Viollet and a final undergraduate report by Fabrice Isart were produced [WVIR00]. Viollet implemented a real-time control of CHANT in Max/MSP using the WACOM tablet, while Isart studied the ergonomics of the tablet when used as a tool for the simulation of a cello bow [Isart99].

- Pedagogy

Many activities related to gestural control have been carried out by the Pedagogy Department, under the directorship of Marie-Hélène Serra (Jean-Baptiste Barrière until May/97). These consist of support to composers interested in this area, courses proposed for different audiences and the coordination of trainees working closely in relation with the composers.

- Compositions

Young composers are accepted each year as part of the Cursus de Composition', a one year course on computer music. During the last years, three compositions have been created using different sensors and/or gestural interfaces: Suguru Goto - Virtual AERI, Lucia Ronchetti - Eluvion-Etude, and Roland Auzet - OROC.PAT and Le Cirque du Tambour.

- Courses

Also regarding the Pedagogy department, a regular course on sensors and gestural capture is given yearly. They may be part of the ATIAM MSc. or the cursus de composition (one-year long), or offered as weekend courses. Lecturers included C. Cadoz, M. Waisvisz, B. Rovan, Emmanuel Flety, Benjamin Thigpen and Marcelo Wanderley. MSc Course on Multimedia at CNAM In the school years 1998-1999 and 1999-2000, the Pedagogy Department at Ircam has taken part in the MSc. course at the Conservatoire National des Arts et Metiers offering a week-long course on Sound and Interactivity. This course deals with different subjects such as sound synthesis, interactive synthesis, new instrument design, gestural control, etc. Apart from Ircam lecturers, invited speakers included, among others, Sally-Jane Norman, Michel Waisvisz, Robin Bargar, Insook Choi, and Robin Minard.

- **Research**

Emily Morin and cellist Benjamin Carat developed a joint project on the measurement of cellists bowing techniques by developing a sensor (FSR) glove and measuring different performances of various performers [Morin2000].

- **Session and Workshop at ISEA 2000**

Marie-Helene Serra and Marcelo Wandeley organized a special session at ISEA2000: Towards a Descriptive Approach of Gesture and Sound Interaction. It consisted of four lectures by researchers Marcelo Wanderley and Antonio Camurri and percussionist Roland Auzet and composer Yan Maresz. They were joined by choreographer Francois Raffinot in a round table coordinated by Marie-Helene Serra. Also during ISEA2000, Emmanuel Flety and Benjamin Thigpen gave a workshop on sensors and interactive systems.

- **Hardware Developments**

Engineer Emmanuel Flety has developed several devices these last years at Ircam. These include the AtoMIC Pro I and II Analog-to-MIDI interface, a ultrasound sensor [Flety00], and an infrared sensor used for interactive dance pieces by the Choreography Department recently created at Ircam.

- **Dance**

The Choreography Department, headed by Francois Raffinot, was created in 1999. The first interactive piece produced is a collaboration between Raffinot and composer Yan Maresz called Al Segno. It uses an infrarouge sensor frame of 2 by 2 meters developed by Emmanuel Flety during one of its movements.

3. Other Activities

Discussion Group - Gesture Research in Music Homepage The Groupe de Discussion sur le Geste Musical was created in 1997 by composer B. Rovan, and researchers M. Wanderley and S. Dubnov. The group's main activities, apart from the maintenance of the site, relate to the development of a bibliography on the subject and the organization of regular meetings. Apart from its internal site, an external site called Gesture Research in Music contains various links and information to different researches and developments related to gestural control.

Meetings and Invited Lectures

Meetings may be either organised as discussion meetings on a pre-determined theme or may consist of an invited lecture by a researcher in this field.
Lectures have been presented by David Wessel (CNMAT), Axel Mulder (Infusion Systems) and Mark Goldstein (Buchla and Associates), Christophe Ramstein (Haptic Technologies, Canada) and Dinesh Pai (University of British Columbia, Canada).
Other invited lecturers include Robin Bargar, Insook Choi, Atau Tanaka, Camel and Steve Coleman, Cadoz, Steim, Robin Minard, Sally-Jane Norman, Michel Waisvisz, among others.

K-Box : programmation d'une extension logicielle pour instrument percussif

Karim BARKATI
CICM - Université Paris 8
barkazik@club-internet.fr

Résumé : Cet article présente d'abord le logiciel K-Box, un instrument informatique sensible à l'intensité des attaques percussives et uniquement à ce critère, puis brièvement l'opéra *Richter, un opéra documentaire de chambre* pour lequel il a été développé, en collaboration avec son auteur, Mario LORENZO. Il décrit ensuite techniquement les algorithmes utilisés et leur organisation au sein du programme, en exposant les raisons des choix effectués au cours du développement de ce logiciel, sous l'environnement de programmation Max/MSP. Il propose enfin quelques axes de réflexion qui me paraissent fondamentaux en ce qui concerne la programmation d'outils numériques pour les compositeurs : la collaboration entre programmeur et compositeur, la notation en rapport avec ces nouveaux outils, le degré d'abstraction nécessaire à l'acte compositionnel, et le problème de la pérennité de ce type d'œuvres.

Mots-clés : interface logicielle, exécution musicale, temps réel, méta-instrument.

1. Introduction

Le développement du logiciel présenté ici a entre autres été motivé par la problématique suivante : plutôt que d'utiliser un clavier ou de construire une nouvelle interface de contrôle pour le temps réel, comment utiliser directement l'instrument du musicien ? En effet, un instrumentiste maîtrise son propre instrument bien mieux qu'une interface spécifique, et jouerait donc plus précisément une partie synthétique contrôlée directement par son instrument [1]. Par ailleurs, la grosse caisse ne permet de considérer que les nuances, paramètre à la fois hautement musical et plus aisé à détecter que la hauteur par exemple. Cette stratégie directe présenterait cependant un inconvénient majeur : la grosse caisse est un peu bruyante en tant que simple interface de contrôle, sauf quand le compositeur souhaite lui laisser son statut d'instrument acoustique pour profiter d'un nouveau rôle double. C'était précisément l'intention de Mario LORENZO dans son œuvre *Richter, un opéra documentaire de chambre*, dans le cadre de laquelle a été développé le logiciel K-Box, grâce à une intense collaboration. Ce logiciel apporte une dimension électroacoustique en sus de l'instrumentarium acoustique, par un système de reproduction d'échantillons en réponse au jeu de la grosse caisse. L'ensemble constitué de cette grosse caisse et du dispositif électroacoustique est dénommé le « Sol Cantante », ou « Soleil Chantant » en français. Nous présentons ici les fonctionnalités et la programmation du logiciel K-Box, afin de montrer les concepts, principes, et notions nécessaires à la programmation d'outils logiciels pour les compositeurs, et les intérêts compositionnels de tels outils.

Après un résumé *Richter, un opéra documentaire de chambre*, la première partie de cet article décrira l'architecture de ce logiciel, c'est-à-dire l'organisation technique des traitements informatiques et audionumériques ; l'environnement de programmation Max/MSP est propice

à ce type d'analyse, grâce à ses orientations graphiques et modulaires. La seconde partie mettra en évidence les méthodes de programmation impliquées dans ce type de développement logiciel, afin de dégager certains principes qui me paraissent fondamentaux.

Résumé de *Richter, un opéra documentaire de chambre* :

En 1948, le physicien autrichien Ronald Richter persuade le général Perón qu'il est capable de maîtriser la fusion nucléaire. Le président argentin, sensible au prestige des savants du Troisième Reich, accueille avec enthousiasme ce projet fantastique : reproduire dans un laboratoire le fonctionnement du soleil. La petite île Huemul, dans un coin paradisiaque de la Patagonie, est transformée en véritable bunker, où vont avoir lieu de mystérieuses expériences. Bientôt, l'Argentine annonce au monde la solution de tous les problèmes énergétiques de l'humanité. Les résultats, cependant, se font attendre, et Richter commence à agir de manière extravagante. Un groupe de scientifiques essaye de prouver que le soi-disant savant est un incapable et un mythomane. Leur chef, le docteur Balseiro, lui rend visite sur son île. Il y trouve un homme à la dérive dans un espace sombre et vide, qui prétend atteindre la température du soleil grâce aux sons produits par une série de haut-parleurs. Richter sera déclaré fou par un psychiatre et son centre de recherche tombera en ruine, mais ses appareils acoustiques seront récupérés par les créateurs du premier laboratoire de musique électronique d'Amérique latine.

Ces faits constituent la base de *Richter, un opéra documentaire de chambre.* Les expériences acoustiques du « savant », rapportées par des documents historiques, y sont développées dans le sens d'une parabole, avec, au centre, un instrument imaginaire : *El sol cantante*. Certains homonymes de Richter – notamment, le pianiste et l'inventeur de l'échelle sismologique – sont convoqués pour la construction du personnage principal. Celui-ci, un baryton, évolue en contrepoint avec son interprète d'espagnol, un rôle féminin, Balseiro, un ténor, ainsi qu'avec douze chanteurs qui tantôt agissent comme un chœur, tantôt comme des personnages secondaires. La partition compte en outre deux pianos, un dispositif de percussion et, bien entendu, des moyens électroacoustiques. L'œuvre dure une heure et quart environ, et se déroule dans un seul espace scénique.

2. Un programme temps-réel basé sur la détection d'attaques percussives

2.1. Architecture du logiciel

2.1.1. Présentation générale

K-Box est un logiciel d'informatique musicale. Plus précisément, il s'agit d'un déclencheur de sons échantillonnés [5], basé sur un système de détection et d'analyse en temps réel d'attaques de type percussives, et doté d'une polyphonie à cinq voies. Il « écoute » le jeu d'un instrument de percussion, et, en fonction de l'intensité de chaque attaque, il déclenche différents sons échantillonnés chargés en mémoire vive. Dans le contexte de l'opéra *Richter, un opéra documentaire de chambre*, on peut se représenter le logiciel K-Box et son attirail électronique comme une excroissance difforme de la grosse caisse, une sorte de greffon artificiel qui a besoin de la grosse-caisse pour s'exprimer, mais qui parle avec d'autres mots. Ce « symbiote » serait un double sensible et multiface, possédant autant de langues que le compositeur-créateur lui en donne. La version 1.61 du logiciel K-Box était suffisamment

stable pour faire nos premières répétitions avec une percussionniste professionnelle : Françoise RIVALLAND a apporté une grosse caisse de taille importante, et nous avons travaillé à définir les limites du système. au niveau des modes de jeu, des baguettes, des vitesses d'exécution, de la position du microphone, et des temps de réponse. Je tiens à la remercier, ainsi que Mario LORENZO pour son professionnalisme et son amitié.

Le logiciel se situe dans une chaîne matérielle précise, dont la grosse caisse et les haut-parleurs constituent les extrémités. Cet ensemble forme ainsi un nouveau corps vibratoire complexe, un nouvel instrument. Différents mécanismes de transduction [3] et de traduction sont mis en jeu à cet effet. La configuration matérielle se compose au minium des éléments suivants : un microphone, un convertisseur analogique - numérique (CAN), un ordinateur Macintosh (à partir des modèles G3), un convertisseur numérique - analogique (CNA), et deux haut-parleurs, ou un système de diffusion plus complet. Le son circule à travers ces éléments sous trois formes différentes : acoustique, électroacoustique, et audionumérique. Il passe d'un état à un autre grâce aux différents tranducteurs et convertisseurs impliqués dans cette chaîne. La grosse caisse n'est pas censée être amplifiée, car l'auditeur doit percevoir le son naturel de l'instrument directement, et n'entendre à travers les haut-parleurs que le son du « méta-instrument ».

Voici la fenêtre principale, qui permet essentiellement d'avoir un accès direct aux paramètres critiques, afin de pouvoir intervenir rapidement en situation de concert :

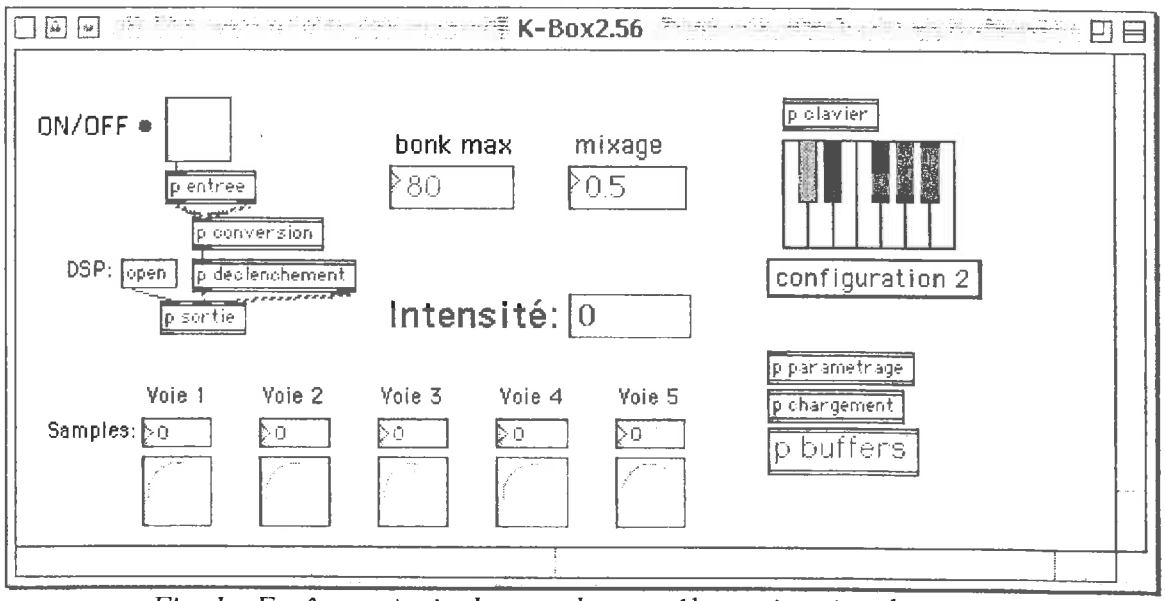


Fig. 1 : Fenêtre principale pour le contrôle en situation de concert

Le schéma suivant reprend le circuit placé en haut et à gauche de la fenêtre principale en indiquant la hiérarchie des patches, et en précisant les objets qui ont un rôle clé au sein de la chaîne de traitement. Sa simplicité et sa totale séquentialité garantissent la reproductibilité du programme.

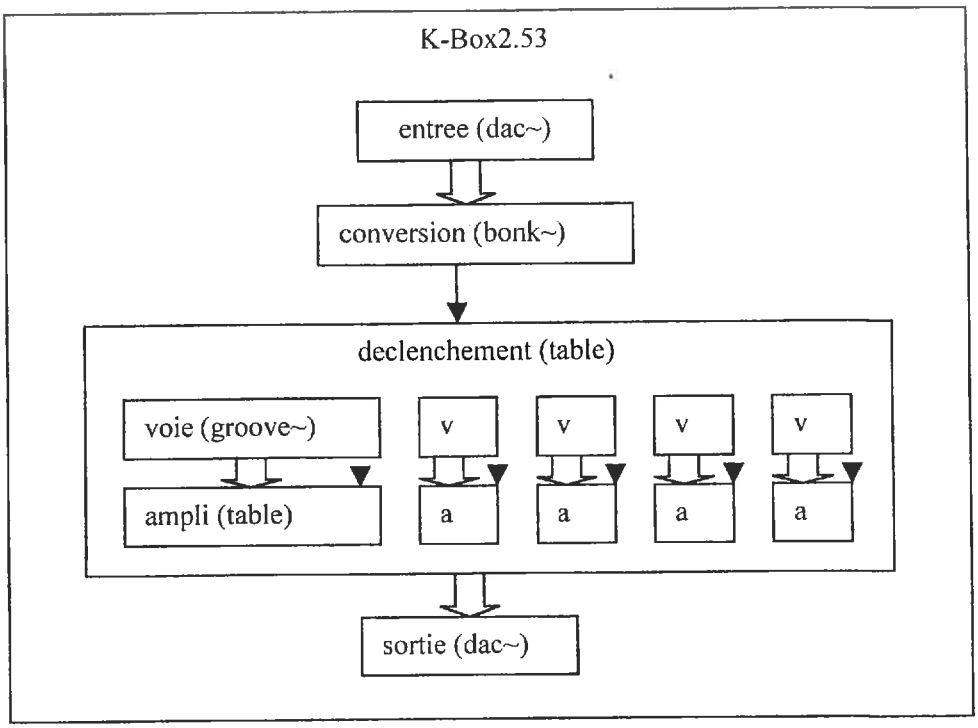


Fig. 2 : Organisation hiérarchique des modules

Les flèches épaisses indiquent la circulation de signal audionumérique, avec un débit constant de 44100 informations par seconde, alors que la flèche noire indique la transmission d'une information non audio, à un débit beaucoup plus lent et non systématique. Le parcours logique du signal s'effectue de manière totalement déterministe et séquentielle.

2.1.2. Description des algorithmes

Le logiciel K-Box s'articule autour des notions d'attaque et d'intensité d'attaque. Chaque attaque détectée dans le signal audionumérique est mesurée dans son amplitude, puis codée parmi 128 valeurs possibles. C'est cette information que nous appelons ici « intensité ». Dans cette version du logiciel, cette information contrôle deux traitements audionumériques : la lecture d'un sample prédéfini, et l'amplification de celui-ci. Nous décrivons dans cette partie à la fois les algorithmes propres au fonctionnement du circuit dépendant de l'intensité, et ceux qui relèvent des autres aspects contextuels à cette chaîne essentielle.

2.1.2.1. La détection et la conversion des attaques

Le patch conversion reçoit un signal audionumérique quelconque, et le convertit en temps réel en une suite discontinue de valeurs résultant de l'analyse de ce signal. Le seul objet bonk~ suffit à détecter les attaques et à mesurer l'intensité de celles-ci le cas échéant. Le reste du patch sert à paramétrer cet objet, afficher différentes valeurs en situation d'ajustement, et à niveler les mesures d'intensité entre 0 et 127.

L'objet bonk~ a été programmé par Miller Puckette, puis porté sur MSP par Ted Apel ; la version 1.1 que nous utilisons date de décembre 1999. Cet objet détecte les attaques des instruments percussifs ; elles sont définies par un changement de forme dans l'enveloppe spectrale. Optionnellement, il peut comparer une attaque avec quelques modèles stockés, pour essayer de deviner de quel instrument provient l'attaque. Cependant, les résultats de cette technique ne sont pas fiables. La description théorique de son fonctionnement est disponible à l'adresse http://www.crca.ucsd.edu/~msp. Les deux sorties de bonk~ sont d'une part le spectre de l'attaque, sous la forme d'une liste de onze nombres correspondant à l'énergie du signal dans les onze bandes de fréquence utilisées, et d'autre part une seconde liste précisant le numéro de l'instrument s'il a été identifié, et surtout la « vélocité ». C'est cette unique valeur qui nous intéresse. Elle s'appelle ainsi car l'intensité de l'attaque, pour les percussions, est directement proportionnelle à la vitesse de la frappe. L'analyse de bonk~ est effectuée sur une fenêtre de 256 points, soit 6 millisecondes pour une fréquence d'échantillonnage de 44100 Hertz, et répétée tous les 128 échantillons, soit toutes les 3 millisecondes.

2.1.2.2. Le déclenchement des samples

Lorsque l'information d'intensité a été codée, tout type de traitement est envisageable, en établissant une correspondance entre les valeurs d'intensité possibles et le domaine de définition du traitement à effectuer. La seule limitation réside dans les 128 valeurs de départ. Le premier traitement consiste ainsi à déclencher des samples préchargés en mémoire, dans un ordre préétabli par le compositeur dans une table de correspondance. La fenêtre du patch declenchement peut être utilisée comme aide à l'apprentissage de l'instrument chargé dans le K-Box. La table de correspondance est dessinée par le compositeur et ajustée en répétition. Elle attribue une unique valeur, entre 1 et 99, à chacune des 128 intensités possibles. Cette valeur correspond au numéro du sample que le compositeur souhaite voir déclenché pour une intensité précise.

Un percussionniste ne possède généralement pas une précision suffisante pour maîtriser 128 nuances différentes. Il suffit pour s'en convaincre d'essayer d'obtenir consécutivement plusieurs fois la même valeur, puis sa voisine immédiate. D'autre part, le système dépend fortement des conditions de captation de l'instrument acoustique : qualité et placement du microphone, réglages de la table de mixage, bruit dans la salle, etc. Toutes ces imprécisions sont minimisées par les diverses possibilités d'ajustements mémorisables du logiciel, mais pas suffisamment pour supprimer un phénomène d'incertitude non négligeable. Cette incertitude conduit à la notion de zones perceptives d'intensité. Nous avons choisi d'en concevoir cinq, par exemple *ppp*, *p*, *m*, *f*, et *fff*, qui forment un compromis entre la précision maximum de l'instrumentiste après apprentissage, et la précision minimum de l'ajustement des paramètres. Il y a donc cinq voies qui correspondent à ces cinq zones, et chaque voie peut accéder à vingt samples : de 1 à 19 pour la première, de 20 à 39 pour la deuxième, de 40 à 59 pour la troisième, de 60 à 79 pour la quatrième, et de 80 à 99 pour la cinquième (fig. 4). Des développements futurs pourraient porter sur des modifications de ces paramétrages, comme le changement de vitesse ou le bouclage. Un objet meter~ (VU-mètre) permet de contrôler visuellement la bonne marche de la lecture.

2.1.2.3. L'amplification des samples

L'amplification donne une dimension plus réaliste à la perception. C'est aussi, dans le cadre de cette recherche, un exemple extrapolable de traitement audionumérique basé sur l'unique information d'intensité. L'activation du traitement d'amplification repose sur le même principe que le déclenchement des samples : l'un des cinq patches ampli est choisi d'après le numéro du sample correspondant à l'intensité reçue, puis ce patch traite directement l'information d'intensité. L'amplification se fait alors simplement par multiplication du sample tel qu'il est enregistré en mémoire par le coefficient déduit de la table (fig. 5 et 6). Un objet r antiClic paramètre à distance la durée d'amortissement lors d'un changement brusque de coefficient, grâce à l'objet line~, afin d'éviter de produire des « clics » numériques.

2.1.2.4. La gestion des configurations

Le caractère multiface du logiciel K-Box provient de la possibilité de charger de nouvelles données en situation de concert. Chaque configuration contient à la fois les paramétrages et les nouveaux sons. Il est prévu que l'instrumentiste puisse changer de configuration à distance, à partir d'un contrôleur MIDI. En haut et à droite de la fenêtre principale apparaissent un clavier pour visualiser la note envoyée par le contrôleur, et un menu déroulant pour indiquer le nom de la configuration qui correspond à cette note. Le changement de configuration repose sur des systèmes de chargement de fichiers. Ces fichiers sont de quatre types :

- les configurations
- les tables
- les instruments
- les fichiers audionumériques

Le logiciel K-Box a été pensé comme une boîte à instruments, où un seul instrument peut « chanter » à la fois. On peut penser à un comédien qui change de personnage ou de masque au cours de la pièce, mais ne joue qu'un seul personnage à un instant donné. L'identité de l'instrument dépend beaucoup du potentiel de singularisation des samples qui le composent. De plus, les auditeurs identifient auditivement la « boîte » elle-même, en tant que processus réactif aux attaques percussives. Un changement de configuration entraîne un changement d'instrument, ce qui constitue sans doute la partie la plus perceptible, mais aussi un changement du paramétrage du logiciel et des deux tables d'association.

Le K-Box peut être défini comme un « méta-instrument », afin de rendre compte qu'il s'agit d'un instrument spécial [6]. Après quelques minutes de pratique se développe une conscience musicale du contrôle de cet instrument informatique, malgré le fait que ce contrôle soit indirect. Ce dernier point justifie l'emploi du préfixe « méta ». Quelques autres aspects renforcent cette idée d'abstraction, comme la vacuité latente du logiciel tant qu'une configuration n'est pas chargée, ou la possibilité d'une notation spécifique exhaustive. Enfin, le sens étymologique « après » correspond tout à fait au mécanisme caractéristique du K-Box, qui réagit *d'après* un signal émis, et *après* lui dans le temps, fût-ce de quelques millisecondes, à cause du temps nécessaire à l'écoute, aux conversions, et à l'analyse de ce signal externe.

2.2. La notation

« Il est absolument nécessaire que les compositeurs soient associés à tous les travaux de recherche, de façon à toujours donner le point de vue du musicien et du créateur, ce qui n'a pas toujours été le cas » affirme Tristan Murail dans un entretien avec Danielle Cohen-Lévinas [2]. En effet, cette collaboration est indispensable à plus d'un titre, en particulier en informatique musicale. Les connaissances musicales d'un programmeur, fût-il lui-même compositeur, ne peuvent prétendre se substituer à la richesse d'une collaboration. Le développement du logiciel K-Box nous a confronté dès l'analyse à des problèmes de notation et de représentation, pour pouvoir communiquer efficacement et précisément. Sa nature de boîte, même virtuelle, confère à ce logiciel une certaine gourmandise innée pour les informations à fournir en entrée. Il a donc fallu concevoir un système de notation pour structurer ces données, tel que cette notation soit la plus intuitive possible, et porteuse de sens. Ce système s'articule autour de plusieurs fichiers textuels et graphiques : la configuration, l'instrument, la table de correspondance, et la table d'amplitude. Pour le concevoir, nous avons tenu compte de la relation de ce système avec la partition, et de la propension à l'abstraction qu'il était capable de générer.

2.2.1. Le fichier de configuration

Le fichier de configuration est un fichier textuel précisant les valeurs d'une dizaine de paramètres, et les noms des autres fichiers à charger. Ce fichier centralise toute l'information, permettant ainsi de se représenter totalement un élément de la boîte avec un seul objet conceptuel. Ainsi, lorsque qu'une pièce fait appel à plusieurs configurations, le nom de la configuration à charger peut apparaître directement sur la partition.

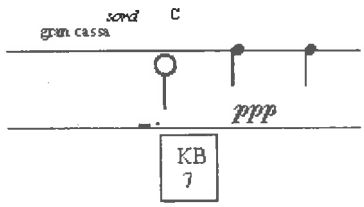


Fig. 3 : Notation d'un changement de configuration sur une partition de grosse caisse

Cette notion de configuration et de changement de configuration fait sens pour le compositeur, car elle correspond à une tradition d'écriture orchestrale de changement d'instrument dans une partition parmi une panoplie de départ, comme un changement de clarinette d'un clarinettiste ou un changement de percussion d'un percussionniste. Plus récemment, cette notion peut faire référence à l'instruction *program change* en MIDI.

2.2.2. Le fichier instrument

Le fichier instrument est aussi un fichier textuel. Il dresse la liste des noms des fichiers audionumériques à charger, ainsi que les numéros des buffers dans lesquels ils doivent être chargés. Cette représentation a l'avantage d'être exhaustive, puisqu'elle précise le nom de tous les samples qui seront chargés pour un instrument. Tout compositeur familiarisé avec les échantillonneurs assimile intuitivement cette représentation à la notion de *banque*. Le fichier

instrument diffère cependant des banques des échantillonneurs au moins par deux aspects. D'abord, les samples ne sont pas affectés à des intervalles de hauteurs (*keygroup*), ni ne répondent à des messages MIDI du type *note-on*. Ensuite, les numéros des buffers sous-tendent la notion de zone perceptive de nuance, qui sont au nombre de cinq dans ce programme. Comme le nombre de samples par zone est variable et que le maximum global est 99 samples, le système de notation permet de repérer les zones d'après le numéro des buffers, en sachant que la première commence à 1, la deuxième à 20, la troisième à 40, la quatrième à 60, et la cinquième à 80.

2.2.3. Les fichiers de tables

Les tables sont des fichiers graphiques, car Max propose des outils d'édition graphique de ces fichiers. Du point de vue strictement informatique, ces tables ne sont qu'une série de couples de valeurs associées, d'où leur nom technique de « tables d'association », mais ne possèdent pas de caractère graphique intrinsèque. L'aspect graphique recèle pour nous un trésor : le compositeur retrouve un outil calligraphique qui lui permet de penser des formes, et donc de retrouver l'abstraction si chère à son art. Une table possède deux dimensions : les valeurs d'entrée en abscisse, et les valeurs de sortie en ordonnée. On peut faire correspondre ainsi pour tout traitement un espace de réaction à un espace de perception. Le logiciel K-Box ne possède qu'un seul espace de perception et deux espaces de traitement, respectivement l'intensité des attaques percussives, codée sur 128 valeurs, le déclenchement d'un sample parmi 99 possibles, et l'amplification d'une zone, codée sur 128 valeurs. Les représentations graphiques des deux tables de correspondance et d'amplification forment des courbes prégnantes pour le compositeur, et superposables verticalement puisqu'elles sont définies à l'identique pour l'intensité en abscisse.

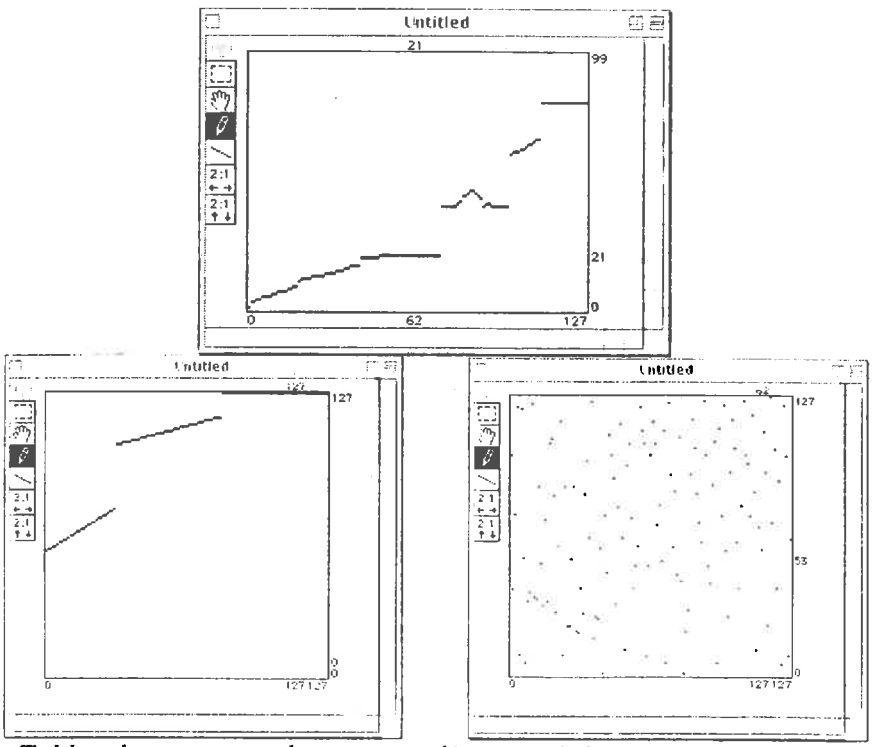


Fig. 4, 5, 6 : Tables de correspondance entre l'intensité des attaques et la liste des samples (fig. 4) ou l'amplification des samples (fig. 5 et 6)

Ces courbes sont directement liées aux samples chargés, et définissent la réponse acoustique du programme, la façon dont il va réagir à ce qu'il « entend ». La notion de zones perceptives exploitée dans le fichier instrument peut être renforcée ou détruite par une courbe. Le compositeur reste maître de ses choix sur la forme de la réaction du programme, et décide du degré de causalité perceptive dans sa notation. La table ci-dessus à droite illustre une volonté évidente de décorrélation.

2.2.4. La partition

La notation traditionnelle convient à cet instrument informatique qu'est le K-Box. Étant donné que le programme ne perçoit que la seule dimension musicale des nuances, la notation rythmique habituelle fait sens, ce qui n'est pas forcément le cas des systèmes plus répandus de déclenchement de samples basés sur les numéros de notes du contrôleur sans corrélation perceptive avec la hauteur du sample. La partition rythmique, sur une portée d'une ligne, n'est en rien mensongère puisqu'elle ne précise pas les hauteurs, et, par contre, elle renseigne sur le type de sample qui va être déclenché si le compositeur a établi un lien entre les nuances et l'identité musicale des samples.

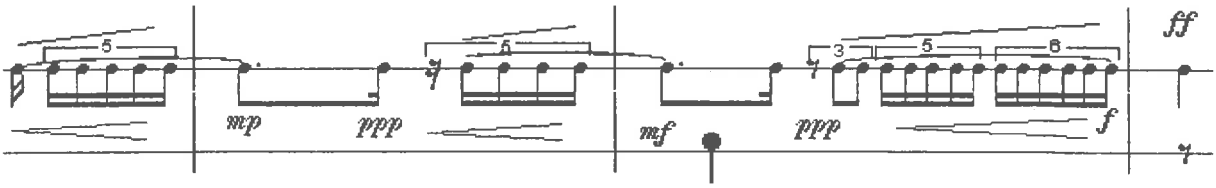


Fig. 7 : Extrait de la partition pour le « Soleil Chantant »

Il n'y a qu'une seule partition pour deux instruments, mais quelques astuces sont envisageables pour modifier l'équilibre, ou le rapport de force, entre le son direct de la grosse caisse et le son diffusé de son double informatique. En effet, on peut paramétrer une configuration pour rendre le programme insensible à certaines baguettes, ou encore modifier la courbe d'amplitude de façon à obtenir une réponse disproportionnée du programme à un jeu percussif pianissimo.

3. Vers la pérennité

Au dire des compositeurs, la pérennité des œuvres qui utilisent du matériel électroacoustique est menacée, voire impossible, notamment à cause de l'obsolence rapide de ce même matériel. Certains compositeurs se sont même résolus à ce que leurs pièces ne soient jouées qu'une seule fois. Les bandes magnétiques se désagrègent, les ordinateurs sont obsolètes dès leur sortie, les programmes ne sont plus compatibles avec les nouveaux systèmes d'exploitation, et les systèmes dédiés disparaissent, en laissant derrière eux des pièces dont la maintenance coûterait un prix exorbitant. La lutherie électroacoustique [4] rencontre des difficultés supplémentaires dûes au temps parfois considérable que représente l'apprentissage d'un nouvel instrument, de surcroît pour jouer une littérature contemporaine souvent difficile techniquement. Dans ce contexte, l'espoir d'une pérennité des œuvres contemporaines utilisant des instruments électroacoustiques semble vain.

Pourtant, nous avons trouvé des formes de solution au travers de cette expérience. Tout d'abord, le contrôle gestuel du K-Box ne nécessite pas d'apprentissage de gestes nouveaux de la part de l'instrumentiste, sinon d'une sensibilité nouvelle. Ensuite, le système de notation élaboré fournit une description exhaustive et relativement intuitive des informations spécifiques à chaque configuration, et à la façon de jouer cet instrument. Les fichiers textuels résistent parfaitement au temps, ainsi que les fichiers audionumériques stockés sur CD-ROM, qui trouveront toujours un logiciel de conversion en cas d'obsolescence du format. Enfin, le logiciel est reprogrammable dans n'importe quel environnement informatique apte à traiter du signal audionumérique, grâce à la description totale de son architecture par le schéma séquentiel suivant :

1. analyse de l'intensité des attaques percussives,
2. codage sur 128 valeurs,
3. déclenchement de la lecture des samples d'après la table de correspondance sur une polyphonie de cinq voies,
4. amplification par voie d'après la table d'amplification,
5. mixage des cinq voies.

La numérisation et le principe informatique de compatibilité ascendante permettent déjà de penser que les instruments informatiques peuvent durer plus longtemps que leurs parents électroacoustiques [7]. Quelques stratégies supplémentaires sont susceptibles d'augmenter leur espérance de vie, en portant la possibilité de dématérialisation à son maximum. Si la partition a été jusqu'à présent le seul lien avec les œuvres du passé, nous proposons de joindre à la partition d'aujourd'hui les informations qui décrivent totalement nos traitements informatiques : à la fois les données, sous forme textuelle, graphique, et audionumérique, et les algorithmes, par un schéma complet. Il devrait être ainsi possible de recréer ces pièces, dans un futur proche ou lointain. En d'autres termes, tout programme fonctionnant à une date donnée est reprogrammable ultérieurement, sous une forme éventuellement différente, et dont on a la certitude qu'il fonctionnera aussi. Il suffit alors que les instruments électroacoustiques se résument à de tels programmes connectés à des appareils triviaux, tels un microphone ou des haut-parleurs, pour qu'il devienne envisageable de recréer les pièces qui utilisent ces instruments, dans un futur proche ou lointain.

Références :

[1] CADOZ Claude, *Le timbre, métaphore pour la composition*, éditions Christrian Bourgois, Ircam, 1991

[2] CAHIER DE L'IRCAM N°1, *Composition et environnements informatiques*, éditions Ircam - Centre Georges-Pompidou, 1992

[3] JACOBSOHN Ricardo E., *Méthode pour l'enseignement des techniques du son*, Université Paris VIII, Département de Musicologie

[4] LALIBERTÉ Martin, *Un principe de la musique électroacoustique et informatique et son incidence sur la composition musicale* École des Hautes Études en Sciences Sociales, 1984

[5] LESBROS Vincent, *Atelier Incrémentiel pour la Musique Expérimentale*, Université Paris VIII, Département d'Informatique, 1992

[6] MACHOVER Tod, *« Classic » Hyperinstruments*, 1992 DOBRIAN Christopher, *The MSP documentation*, University of California, Irvine, 1998

[7] ROADS Curtis, *L'Audionumérique*, version française : Jean de Reydellet, éditions Dunod, Paris 1998

Max et MSP :
http://www.cycling74.com/
Centre de Recherche Informatique et Création Musicale :
http://www.ai.univ-paris8.fr/~vi/cicm/
Institut International de Musique Electroacoustique de Bourges :
http://www.gmeb.fr/
Groupe de Recherches Musicales :
http://www.ina.fr/GRM/
Institut de Recherche et Coordination Acoustique - Musique :
http://www.ircam.fr/
Société Française d'Informatique musicale :
http://www.sfim.org/
Association pour la Création et la Recherche sur les Outils d'Expression :
http://www-acroe.imag.fr/

L'Interpolateur, une interface de contrôle multi-paramètrique

Daniel Teruggi
Groupe de Recherches Musicales, Institut National de l'Audiovisuel
116 Av du Prés. Kennedy 75220 Paris Cedex 16
dteruggi@ina.fr
Martin Spain
Faculty of Engineering and Information Sciences
Université de Hertfordshire
College Lane, Hatfield, AL10 91B
m.spain@herts.ac.uk

Résumé : Les interfaces de contrôle ont souvent imité des modèles inspirés des appareils analogiques du passé. Il est difficile dans ce contexte de contrôler plusieurs paramètres simultanément avec l'accès bi-dimensionnel de la souris. L'interpolateur propose une surface de contrôle dans laquelle des ensembles de données sont représentés par des cercles, permettant des interpolations complexes, basées sur le positionnement géographique des objets avec des champs d'action variables et une automation des déplacements.

Mots clés : Interpolation, interfaces, contrôle, multi-paramétrique, traitement du son, systèmes de contrôle graphique, GRM Tools.

Interfaces de contrôle

Dans le domaine de la manipulation virtuelle de sons, les interfaces jouent un rôle essentiel par leur implication dans le comportement des utilisateurs. Tout logiciel destiné à une utilisation technique ou de consultation, doit répondre à des contraintes bien précises, mais doit surtout proposer une interface d'utilisation adaptée à la fonction et simplifiant les taches les plus usuelles.

Il est habituel dans ce domaine de s'inspirer d'analogies gestuelles issues d'expériences quotidiennes ou dictées par des pratiques manuelles. La virtualité des interfaces a été développée à partir de reconstructions ou simulations des modes de contrôle habituellement utilisés pour contrôler des systèmes physiques. Ainsi, dans le domaine du traitement du son, les interfaces graphiques ont simulé les boutons et potentiomètres linéaires des synthétiseurs et boîtes de traitement du passé, cela leur apportait une immédiateté opérationnelle et une clarté de compréhension.

Cette approche de la représentation a imprégné le développement des interfaces de contrôle, surtout ces dernières années, étant donné la facilité de programmation d'interfaces graphiques et l'argument indispensable qu'elles constituent pour tout produit destiné à une commercialisation. La reconstruction virtuelle d'interfaces physiques présente un problème majeur et c'est celui de l'accès de contrôle. Dans un système physique, plusieurs boutons ou réglettes peuvent être contrôlés par les différents doigts des deux mains ; dans une interface informatique l'accès privilégié reste la souris, ce qui limite le nombre de paramètres contrôlables au même temps.

Les systèmes inspirés de la réalité se révèlent alors efficaces si le nombre de paramètres à contrôler reste peu nombreux ; dés qu'une vingtaine de paramètres sont présents dans un système, leur contrôle par un seul paramètre physique devient complexe et décourageant.

Par ailleurs l'approche multi-paramétrique pour contrôler un système complexe n'est pas toujours la meilleure pour contrôler un système. Les algorithmes de traitement et de contrôle présentent toujours un nombre, plus ou moins important, de variables à contrôler (variables qui vont devenir des paramètres dans le contexte de l'interface). Le résultat de l'algorithme peut ne pas être de type paramétrique ; si l'algorithme propose la gestion d'un nuage d'évènements sonores, contrôler chaque événement individuellement peut s'avérer impossible à partir de paramètres simples. Il faut alors des paramètres complexes ou bien des macro-paramètres permettant un contrôle global du système.

Les deux problématiques principales liées au développement des interfaces sont : comment créer des interfaces inspirées des particularités de l'accès de contrôle que représente la souris, et comment créer des systèmes de contrôle global en présence d'une multitude de variables à contrôler.

Le modèle GRM Tools

Le développement des algorithmes a rendu les paramètres de contrôle de plus en plus nombreux et il a fallu trouver des solutions pour les contrôler à partir de l'accès bidimensionnel de la souris. Parmi les solutions trouvées, outre celle d'éliminer les paramètres pouvant être considérés comme superflus, était celle de grouper de nombreux paramètres identiques en un ou deux paramètres généraux qui permettront soit de les contrôler tous en parallèle, soit d'introduire des facteurs de séparation entre les paramètres de manière à introduire une différenciation progressive.

Cette approche globale du contrôle paramétrique, tout en simplifiant le contrôle, introduit un mode opératoire général. Au lieu de déterminer chaque paramètre individuellement, une fonction conditionne le fonctionnement du système, fonctionnement dont le détail reste invisible à l'utilisateur, et le résultat est une action multi-paramétrique sur le son qui altère globalement sa constitution. Pour un ensemble d'oscillateurs, par exemple, on déterminera l'écart entre chaque fréquence et ainsi on contrôlera toutes les fréquences. En mode opératoire, pour mieux saisir le fonctionnement du système, une représentation graphique de l'action permet de comprendre l'implication de la variation du paramètre.

Les GRM Tools[1] ont été construits autour de cette approche qui cherche à réduire le nombre de paramètres tout en les rendant très puissants en termes de conséquences opérationnelles et expliqués à travers des fenêtres dynamiques qui rendent compte des changements principaux. Le nombre de paramètres par algorithme reste relativement réduit (moins de 20) et la combinatoire riche et extrêmement variée.

Un deuxième aspect concernant les algorithmes réside dans la possibilité de capturer l'état de l'ensemble de paramètres à un instant donné, pouvant être mémorisés dans un « setting » ou mémoire d'état. Plusieurs settings peuvent être crées sur chaque algorithme et rappelés de manière instantanée. Ainsi, une configuration du système est réalisée pour l'adapter aux contraintes de l'utilisateur. Un passage progressif d'un setting à un autre peut être réalisé, à

[1] GRM Tools©, logiciel de traitement développé par l'Ina-GRM destiné aux environnements TDM et RTAS de Digidesign, et à l'environnement VST de Steinberg.

vitesse de changement constante ou contrôlée manuellement ; permettant ainsi de trouver des états intermédiaires entre deux états prédéterminés.

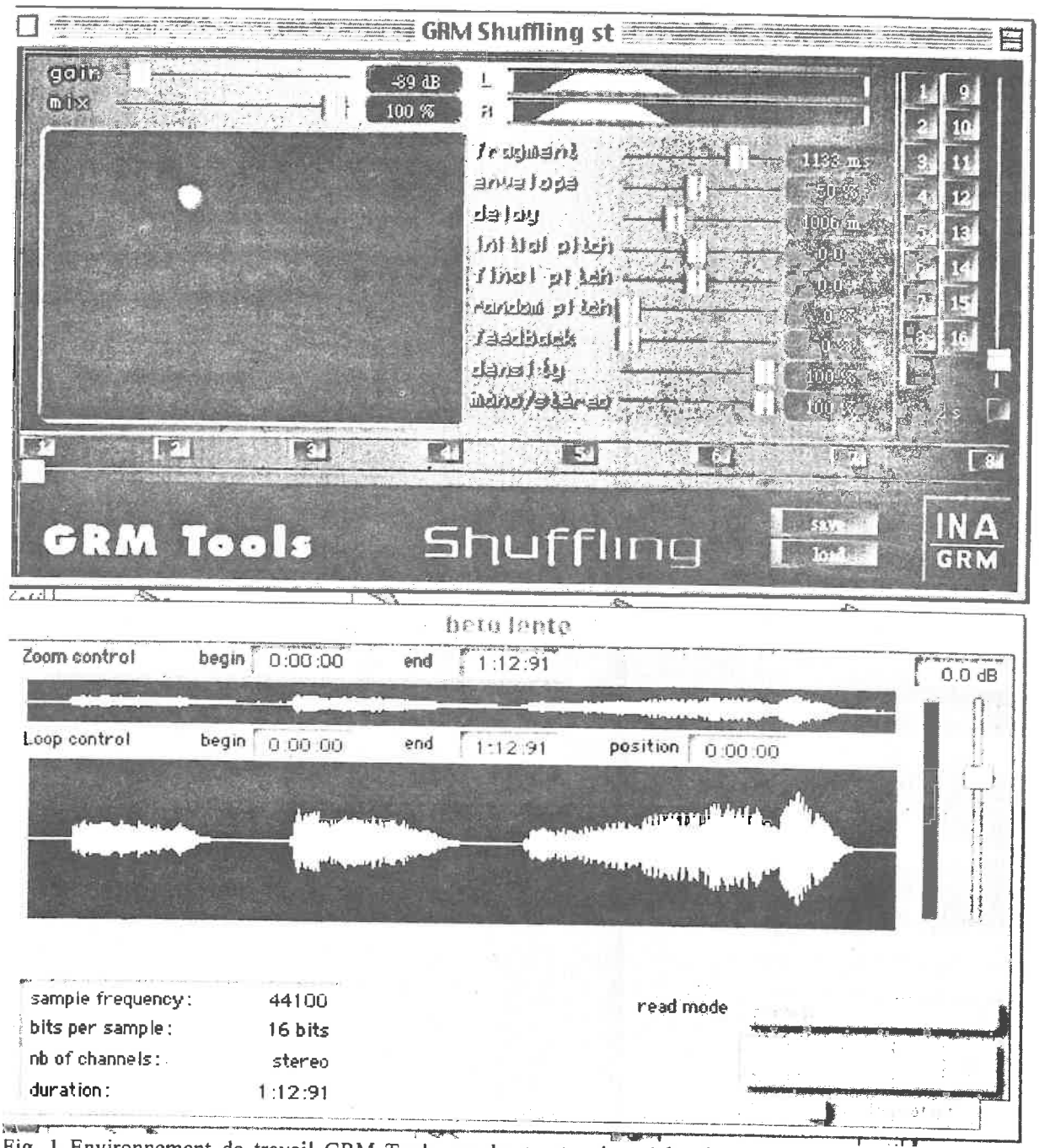


Fig. 1 Environnement de travail GRM Tools, en, haut est présenté le plug-in de brassage *Shuffling* avec l'interface bidimensionnelle de contrôle permettant le contrôle simultané de la durée des fragments et de leur position temporelle (paramètres fragment et delay), les 16 cases à droite représentent les mémoires d'état ou « settings ». Plus bas se trouve le player permettant la lecture et enregistrement des sons.

L'Interpolateur

Un autre niveau de travail consiste à faire fonctionner plusieurs algorithmes simultanément sur un même son. Dans ces conditions, le nombre de paramètres devient rapidement très

important et, chaque algorithme devant être modifié à tour de rôle, le contrôle général devient impossible.

Un projet de recherche a été mis en place avec l'Université d'Hertfordshire[2] à Hatfield en Angleterre, pour développer un outil d'interpolation général pour l'ensemble des algorithmes de GRM Tools. Cet outil permet de mémoriser l'état des paramètres de l'ensemble des algorithmes présents sur GRM Tools et de les placer dans un espace bi-dimensionnel dans lequel des relations de type topographique vont se développer entre ses mémoires, cet espace est appelé espace de contrôle. Un champ d'action est établi autour de chaque mémoire et des interpolations très fines peuvent être réalisées entre plusieurs ensembles de valeurs. Deux types de champ d'action peuvent être établis, les champs circulaires, qui réalisent une interpolation dans toutes les directions à partir du point central, et les champs limités, appelés « projecteurs » qui orientent l'action dans une direction avec un degré d'ouverture et un rayon d'action. Des constructions topographiques très complexes peuvent être construites, les différentes couleurs indiquent des ensembles de paramètres différents associés à cette couleur.

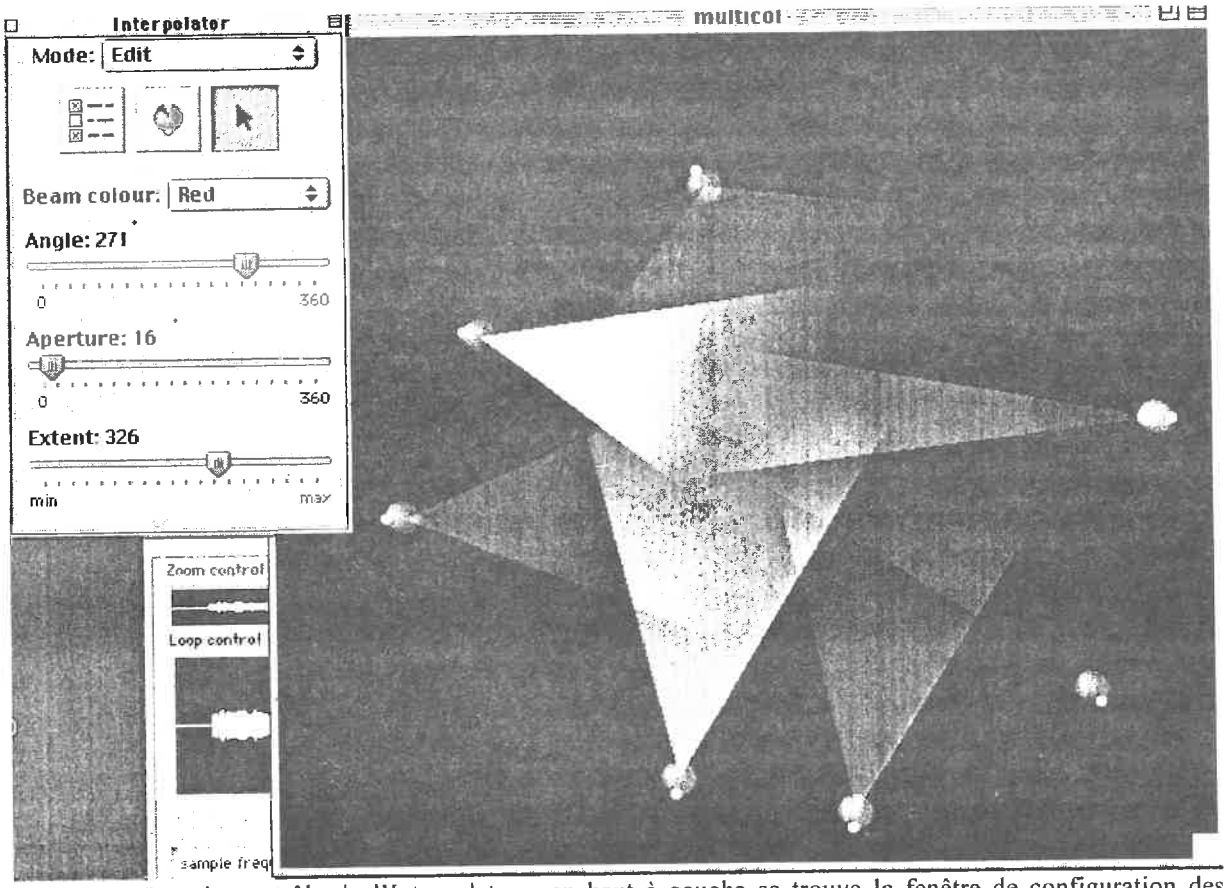


Fig. 2 Interface de contrôle de l'Interpolateur, en haut à gauche se trouve la fenêtre de configuration des paramètres. L'espace central noir où sont placés les projecteurs de différentes couleurs est l'espace de contrôle à l'intérieur duquel se réalisent les interpolations.

Sur l'interface, des ensembles de valeurs des paramètres en provenance d'algorithmes différents de GRM Tools (jusqu'à 4 plug-ins) représentent un état particulier de configuration. Cet état, représenté par un objet graphique constitué d'une sphère grise à partir de laquelle part un rayon de couleur, est placé dans l'espace bi ou tridimensionnel de l'interpolateur (Fig.

[2] L'université d'Hertfordshire est le seul centre en Grande-Bretagne réunissant les disciplines de : Informatique, Musique, Mathématique et Ingénierie électrique, orientées vers la recherche en Science et Technologie.

2). Des parcours contrôlés manuellement par la souris ou automatisés avec un séquenceur, sont alors possibles entre les différents états. Ces parcours vont provoquer des interpolations entre les différents états selon de modes de comportement différents. Si les états sont représentés par des champs circulaires (Fig. 3), alors l'interpolation est de type gravitationnel, et toutes les mémoires interagissent entre elles si leur rayon d'action se croise. Si les états sont représentés par des champs limités ou « projecteurs (Fig.2), alors l'interpolation se réalise à l'intérieur du rayon dégradé. En quittant le rayon du projeteur un changement abrupt aura lieu.

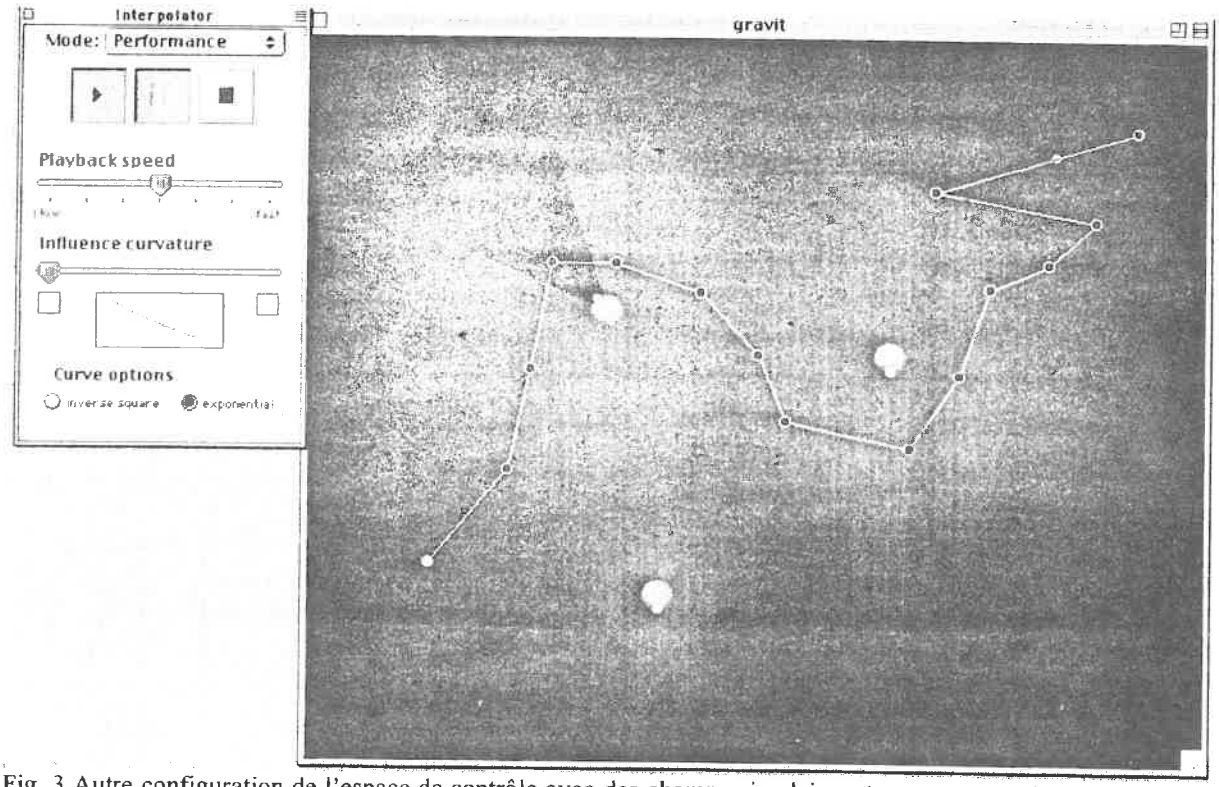


Fig. 3 Autre configuration de l'espace de contrôle avec des champs circulaires et un parcours séquencé qui peut être effectué à différentes vitesses.

L'espace de contrôle (carré noir), de taille variable, est un espace ouvert dans lequel sont incorporées les mémoires représentant les paramètres de différents plug-ins de GRM Tools. Une valeur par défaut lui est attribué avant de placer les différents états ou mémoires (cette valeur par défaut peut être par exemple un état neutre où aucun traitement est réalisé sur le son). Les plug-ins de GRM Tools sont alors configurés selon les critères de l'utilisateur et un état ou mémoire est créée sur l'espace de contrôle ; cet état reprend les valeurs des paramètres des différents plug-ins. Une sélection peut être réalisé de telle sorte à ne capturer que certains paramètres ou bien en ne mémorisant que certains plug-ins. À ce moment-là des mémoires de couleur différentes sont créés (au choix de l'utilisateur) pour ainsi différencier des états qui contiendraient certains plug-ins et pas d'autres. Finalement la largeur du rayon est fixée ainsi que la longueur de son rayon d'action. Une fois les mémoires créées elles sont placées à différents endroits de l'espace de contrôle et l'on peut ; soit les rappeler, soit explorer l'espace existant entre elles. Si des rayons se croisent, une interpolation entre 2 ou plusieurs états se réalisera ; si une mémoire est isolée, l'interpolation se fera entre ses valeurs et les valeurs du fond données au départ.

Pendant l'interpolation, à tout moment, un nouvel état peut être crée. Ce nouvel état va modifier l'équilibre général du système. Un changement de position géographique d'un état apportera aussi un changement dans l'équilibre du système. Le système peut s'utiliser pour réaliser des interpolations entre deux états (passage progressif d'un ensemble de valeurs à un

autre) ou bien comme un espace d'exploration dès que plus de deux états interfèrent entre eux. Dans ce cas, il est possible d'obtenir des résultats tout à fait inattendus, nés de l'interaction entre les différentes valeurs.
D'un point de vue musical, un ensemble mémorisé dans l'interpolateur représente une décision à caractère musical. Une décision est une configuration intelligente d'un système selon des critères d'organisation propres à l'utilisateur ; celui-ci fixe des états des plug-ins et personnalise ainsi le système en fonction de ses objectifs ou désirs musicaux. Plusieurs personnalisations ou décisions sont introduites dans l'interpolateur et un mode de circulation est établi entre les décisions. Le résultat d'une telle action peut revêtir déjà un caractère musical abouti bien que très souvent le résultat soit une pré-organisation de type musical, c'est-à-dire un enchaînement d'événements établis selon les critères de l'auteur. La musique pouvant représenter un état encore supérieur de combinaison d'organisations sonores intentionnelles de ce type.
Concernant le passage d'un ensemble de données à un autre, les interpolations sont relativement faciles à percevoir dès qu'il s'agit de variations de hauteur ou d'intensité, mais elles deviennent difficiles à prévoir dès que les interpolations concernent d'autres types de paramètres dont le comportement est moins évident pour notre perception (des paramètres du type *random*, des paramètres d'écartement de fréquences ou des réinjections ont un comportement souvent difficile à suivre de manière linéaire, surtout s'ils sont combinés). La particularité du travail avec le son, dans lequel la perception d'une évolution ou d'une rupture dépend des phénomènes de perception, font que la linéarité d'une interpolation n'est pas forcément le meilleur modèle de description d'une variation. Très souvent une variation continue apportera un comportement de type "catastrophique" pour notre perception, de ce point de vue, l'interpolateur est un superbe outil pour faire émerger des comportements imprévus, toujours construits autour de la configuration du système réalisée par l'utilisateur en vue de l'obtention d'un résultat à caractère musical.

Une première version du logiciel a été développée par Martin Spain, étudiant en licence à l'Université de Hertforshire, sous la direction de Richard Polfreman. Pour cela une version spéciale des GRM Tools a été développée par Emmanuel Favreau, pour pouvoir permettre le contrôle des paramètres à des ressources extérieures au logiciel. Cette première version a été finie en novembre 2000.

Dans cette première version, il est possible de capturer l'ensemble des paramètres et les organiser dans l'espace bi-dimensionnel de la surface de contrôle. Des interpolations peuvent ensuite se réaliser manuellement entre chaque mémoire et automatiquement à l'aide d'un séquenceur. Pour cette dernière opération, plusieurs points reliés par des traits sont placés dans l'espace et cette trajectoire est parcourue à des vitesses variables permettant de reproduire un parcours et de le modifier progressivement (voir Fig. 3).

Une phase de test est réalisée actuellement auprès des utilisateurs, ce qui permettra de vérifier son efficacité en termes de production et déterminer les améliorations à effectuer. Il faudra par la suite envisager une version intégrée à GRM Tools et étudier les possibilités d'utilisation de l'interface de contrôle sur d'autres applications y compris des applications non liées au contrôle de paramètres sonores.

Bibliographie :

E. Favreau, G. Racot, D. Teruggi, Evolution des outils, évolution des idées, dans Interfaces homme-machine et création musicale, Hermès science, Paris, 1999.

D. Teruggi : Le système Syter, Thèse de Doctorat, Université de Paris VIII, Saint-Denis,1998.

D. Teruggi, L'interactivité dans les processus de création sonore, dans Interfaces homme-machine et création musicale, Hermès science, Paris, 1999.

L'espace perceptif dans la relation du son au geste

Daniel Arfib

CNRS-LMA, 31 chemin Joseph Aiguier
13402 Marseille Cedex20 France
arfib@lma.cnrs-mrs.fr

Résumé: Relier un geste à un son, ou réciproquement peut bénéficier grandement d'une compréhension de l'aspect perceptif d'un son, par la définition d'un espace dans lequel son et geste se projetteront. La définition de ce plan n'a rien de trivial, puisqu'il peut ou non s'appuyer sur une dimension temporelle, mais elle aide à la définition de fonctions de correspondance (mapping) entre son et geste, de stratégies de jeu musical, et permet d'élaborer des systèmes gestuels audionumériques sur des bases compréhensibles.

1. introduction

Comment relier un geste à un son, ou réciproquement un son à un geste[1]? C'est une question qui dans la lutherie traditionnelle se pose souvent en terme de contraintes plutôt que de liberté: le matériau lui-même implique une fabrication d'un instrument qui lui-même impose un jeu instrumental. Par exemple le frottement de l'archet sur la corde s'opère selon des lois précises qui conditionnent le son. Le style de musique aussi, en temps qu'assemblage de sons, conditionne le jeu instrumental et à chaque style on verra se développer une gestuelle particulière. Certes l'habit, au sens du "look" permet de reconnaître immédiatement une image non sonorisée d'un groupe musical, mais en regardant plus finement, il y a une sorte de personnalité du geste, celle des violonistes amorçant un pizzicato, d'un guitariste pop jouant un son distordu, ou d'un luthiste de musique arabo-andalouse.

La dissociation possible du geste et du son, ou plutôt la modularité de cette relation, permise par l'instrumentation électronique, amène en soi la question fondamentale de l'utilisation de la liberté: quelles contraintes donc mettre qui ne soient pas subies, mais acceptées, choisies, recherchées même comme possibilité d'évolution du geste et du son. Cela pose le sujet du but à atteindre: est-ce l'exploration de zones inhabituelles sinon interdites, est-ce la facilité à produire des sons conventionnels?

2. du geste au son, du son au geste

Avant de trouver des solutions pratiques, il est bon de partir des fondements: sur quoi se base la perception d'un geste, sur quoi se base la perception d'un son, et y a-t-il une manière quelconque de trouver des ponts entre ces deux perceptions?

Du côté du geste, la perception peut être du domaine de la sensation : on ressent la pression du doigt sur un matériau par la rétroaction (feedback) nerveuse des cellules placées sous l'ongle. Elle peut aussi être amenée par un autre sens: augmenter le volume d'une console de mixage ne provoque pas (usuellement) un effort supplémentaire, et le retour de la sensation s'effectue par le dosage du son vis a vis du confort, ou de la volonté d'expression liés au volume sonore. Encore du côté du geste, il est une perception interne, pourrait-on dire

[1] Cette étude bénéficie du soutien du Conseil Général des Bouches du Rhône dans le cadre du projet "Le geste créatif en Informatique musicale".

cervicale, du geste, qui fait que le désir de mouvoir un bras est suffisant à sa réalisation (toutefois l'intention de saisir un objet est plus combinatoire puisqu'elle mêle le retour par la vue ou la sensation).

Du côté du son, il est indéniable que l'audition est vraiment le sens primordial, même s'il n'est pas exclusif d'une sensation corporelle pour les sons graves de forte intensité. Mais cette perception auditive peut s'observer selon des couches différentes; le signal sonore lui-même n'est pas perçu par le cerveau : le cerveau perçoit le signal transmis par les cellules ciliées de l'oreille, ce qui veut dire que la représentation du son est déjà biaisée par le prétraitement assez radical de l'oreille interne. Les phénomènes de masquage de fréquences sont de cet ordre, mais plus généralement ces signaux sont déjà "en forme" pour évaluer la hauteur d'un instrument par exemple. Néanmoins cette première couche pour essentielle qu'elle soit, n'est pas suffisante en soi pour expliquer par exemple comment l'on peut percevoir, même sans éducation musicale, l'expressivité d'un geste musical, le phrasé, l'emphase d'une mélodie.

Il y aurait donc un niveau plus complexe où l'intention gestuelle se projette et peut même s'identifier à la perception synthétique du son. C'est une chance en soi, et l'on pourrait résumer cette notion en donnant un exemple: l'instrumentiste provoque dans son corps une intention musicale qui se traduit en un geste, qui donne lieu à un son, aidé en cela par le retour d'écoute et de sensation corporelle. Ce son perçu par l'oreille a été prétraité et transmis au cerveau qui extrait de ces données, en fonction de la sensation et de l'expérience, une représentation du son qui procède de l'intention musicale donnée par l'initiateur du processus. Ce sont les représentations possibles de ce plan intermédiaire entre geste et son, qui sont au cœur de cet article.

2.1 Geste et son, même rythme?

L' espace gestuel et de l'espace sonore n'ont pourtant à priori rien en commun. Les gestes sont dans le domaine dit des fréquences "haptiques", soit quelques Hertz: on peut décrire les coordonnées d'un objet pris en main, et tracer son évolution dans le temps, et les gestes de saccade n'iront pas plus loin qu'une dizaine de Hertz. Cela est constaté dans le geste musical du vibrato, que ce soit sur un instrument à cordes, ou sur un thérémine. Le son, tel qu'il est capté par l'oreille utilise des fréquences bien plus élevées, de l'ordre de 20 Hz à 20 KHz. C'est ici que la différence "traditionnelle" entre matière et forme dans le matériau sonore intervient : la forme du son, par exemple son enveloppe, ou l'évolution de sa fréquence fondamentale, si elle existe, est du domaine haptique. La remarque suivante est donc importante : il y a dans la perception sonore une faculté d'extraire des phénomènes haptiques dans un signal lui-même à fréquences élevées. Cela est très net au niveau du vibrato: la perception sonore d'une modulation lente de fréquence passe du glissé de fréquence (l'oreille suit l'évolution de cette fréquence) à la sensation indifférenciée du vibrato pour des valeurs de 5 à 10 Hz avant de donner la sensation globale d'un timbre, harmonique ou inharmonique (tel qu'il est utilisé par les synthétiseurs FM). Dans un autre registre, les attaques sont perçues globalement, et la richesse typique de leur contenu fréquentiel est absorbée par la sensation en des notions plus globalisantes. Ce sont donc ces phénomènes qui sont les plus notoires dans l'écoute musicale : la sensation de timbre revient à étudier l'évolution de macro-paramètres en fonction du temps, ou même à leur intégration au cours du temps. Dans le premier cas, on percevra le son comme une évolution (comme par exemple une diphtongue en parole: ou-i. Dans le second cas, on peut percevoir une note de clarinette comme une note de clarinette, dont l'unité est pourtant loin d'être évidente au vu du signal physique qui la représente.

2.2 Fonctions de correspondance (mapping) entre geste et son

Coupler un son et un geste en permettant d'exprimer une intention musicale - parfois décrite comme le "geste musical"- est donc relier deux mondes distincts par l'intermédiaire d'une interface. Il est possible, et cela a été fait de nombreuses fois, d'associer des données gestuelles et de données de synthèse sonores à l'aide de fonctions de correspondance entre ces paramètres objectives du geste (position, vélocité d'un objet par exemple) et les paramètres objectifs d'un son de synthèse (par exemple index de modulation d'une synthèse FM). Ces fonctions de correspondance (mapping) ont été classifiées en diverses catégories selon qu'elles font appel à un ou plusieurs paramètres [22].

Toutefois ce couplage est grandement aidé si l'on définit un espace intermédiaire, qui se relie d'un côté au geste et de l'autre au son. Le problème qui est le notre est donc celui-ci: le geste peut-il être codifié dans un espace qui est directement assimilable avec un espace sonore.

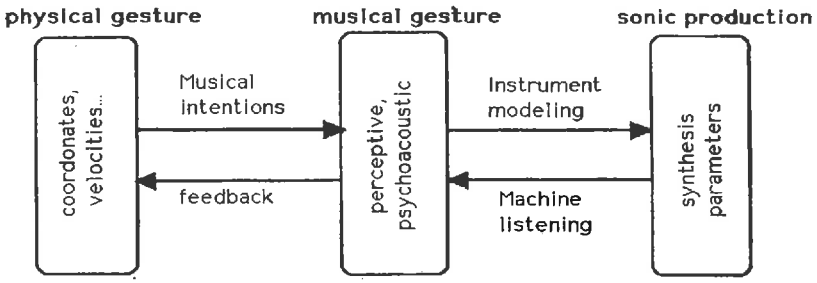


Fig. 1. Le lien entre les gestes physiques et la production sonore peuvent être vus comme un processus double d'action et de retour, et un plan intermédiaire peut être défini, qui représente un espace perceptif ou psychoacoustique.

On trouve dans la littérature des références correspondant à l'existence d'un plan intermédiaire, sous différentes appellations, qui aide au couplage, à la correspondance (mapping) entre données gestuelles et données sonores[1, 10, 17]. Par exemple dans la figure 1 [14], le point principal est la capacité d'extraire des données perceptuelles pour les mettre dans un espace psychologique qu'il soit possible d'associer à des gestes. Cet espace peut être psychophysique ou symbolique.

Ce plan lui-même en fait a deux faces: d'un côté le geste, et d'un autre le son, et l'idéal serait de trouver un plan ou un point sur une face se retrouve sur l'autre face, autrement dit un espace perceptif où perception sonore et représentation gestuelles se confondent. Pour cela, il faut essayer de distinguer ce qu'on peut appeler un espace perceptif.

3 Exemples des représentations d'espace perceptif.

Nous allons ici donner quelques exemples qui vont permettre de voir la diversité et la complémentarité d points de vue sur l'espace perceptif.

3.1 Représentation d'un "espace de timbre de sons isolés"

En présentant à un auditeur des sons isolés, et selon une méthode d'évaluation, on peut classer les sons selon une géométrie perceptive. La méthode des paires de sons consiste à évaluer la ressemblance entre deux sons isolés, et avec l'unique réponse de ces degrés de ressemblance des méthodes statistiques, dont l'analyse multidimensionnelle, permet de dresser une carte géographique de la perception globale de ces sons. Pour prendre un parallèle l'estimation par un automobiliste des distances entre deux villes permet de retracer la carte approximative de la

France. La différence est bien entendu que la carte des timbres est multidimensionnelle, et que sa réduction à deux dimensions ou trois dimensions ne donne qu'un aspect, mesuré par la variance associée, limité de l'observation.

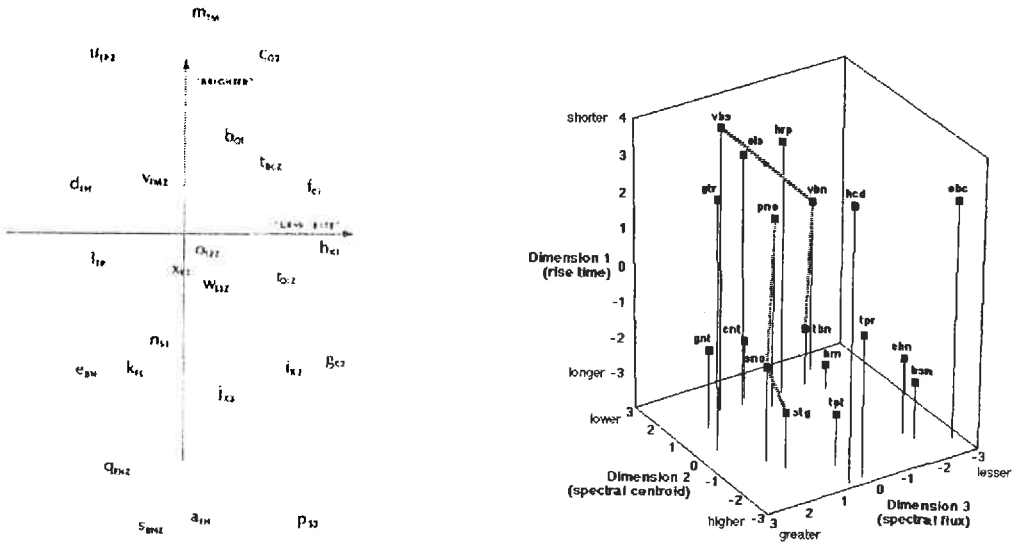


Fig. 2 deux représentations très classiques d'une analyse multidimensionelle à partir de réponses subjectives d'auditeurs à l'écoute de sons isolés [18, 11]

La représentation donnée par Wessel et Grey [9, 16, 18] est désormais un classique de l'Informatique musicale. Son interprétation nécessite de bien évaluer les prémisses de cette expérience: les sons sont isolés, c'est à dire ce sont des notes uniques, et de plus normalisées (sons brefs de même longueur et de même hauteur). Elle touche toutefois à un concept radical: la sensation d'une note unique lors d'un événement long en terme de sensation (de l'ordre d'une seconde). D'autres formes de représentation [11] (par exemple la représentation en clusters) existent, qui pointent sur les différentes régions d'un plan selon une structure hiérarchique.

3.2 Représentation de l'espace des voyelles

Une autre façon de "voir" le son, est de capter son identité présente, et de voir son existence comme une évolution dans le temps. Cela est plus vraisemblable avec des sons évolutifs comme une articulation vocale, notamment par l'enchaînement de voyelles, qui peut conduire au chant diphonique. D'une manière générale, l'évolution de l'enveloppe spectrale d'un son est passible de cette observation.

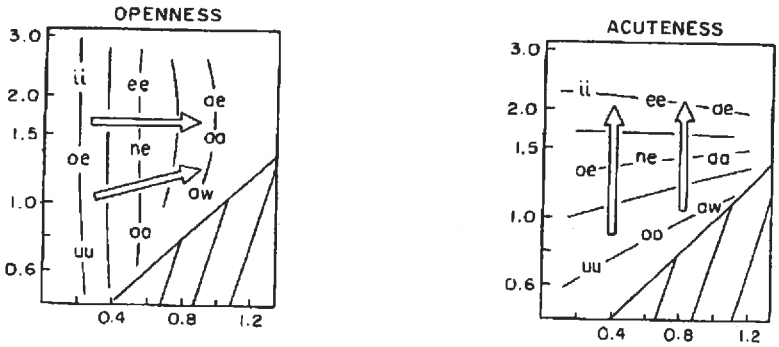


Fig. 3. Dans le plan constitué par les deux formants principaux des voyelles il est possible de représenter les composantes principales de la perception des voyelles [20]

Dans le cas des voyelles, une représentation "psycho-physique" a souvent été donnée par la position sur un plan des fréquences de deux premiers formants (bosses de résonance) des voyelles. D'autres étude, notamment celles de Slawson [20], ont utilisé des méthodes statistiques pour évaluer des espaces perceptifs associés à ces voyelles

Il est radical de comprendre la distinction entre le fait de concevoir un son comme une évolution temporelle et comme un phénomène global. En fait il s'agit de deux écoutes différentes, toutes deux valables : rien n'interdit de dire qu'un son de flûte de deux secondes est l'évolution d'une couleur sonore, ou a contrario qu'elle est une sensation unique qui est catégorisée selon un vocabulaire.

3.5 les trajectoires dans des espaces multidimensionnels

Une méthode d'analyse en composantes principales [7, 21], cette fois effectuée sur le signal lui-même, permet de décrire des trajectoires dans un espace à n dimensions. Tout autant que la trajectoire, c'est la nature des axes qui va donner sa couleur au son.

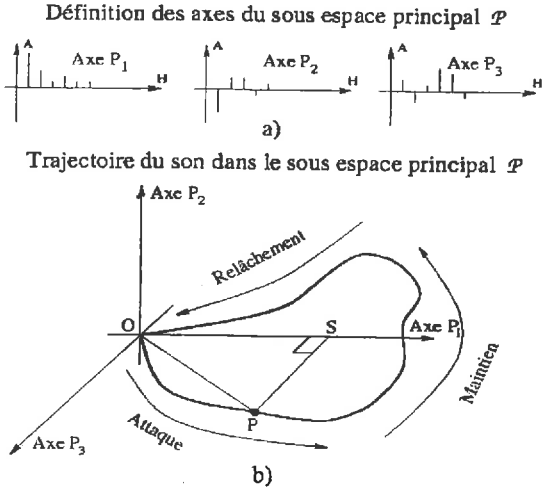


FIG. 2.9 – *Axes et trajectoire*

Fig. 4. Dans un espace tridimensionnel issu du signal et non de jugements subjectifs, on peut tracer l'évolution au cours du temps pour un son unique (et non plus un point unique et symbolique du son isolé) . Figure issue de la thèse de Thierry Rochebois [21]

D'une manière générale l'extraction de paramètres perceptifs permet de construire de telles trajectoires. A titre d'exemple l'extraction de l'amplitude et du centre de gravité d'un son peut permettre des imitations synthétiques par distorsion non linéaire [2] ou par synthèse FM [8] ayant les caractéristiques propres du son original à l'aide des trajectoires réalisées.

3.4 Les interpolateurs "à la Syter"

Un autre exemple de représentation de l'espace des timbres est donné en pratique par l'instrumentation numérique Syter: un ensemble de paramètres est associé à un point d'un espace bi-dimensionnel, et d'autres points, ainsi que leur zone d'influence sont représentés de même. L'évolution d'un pointeur, ici sur une tablette graphique va provoquer l'évolution de ces paramètres.

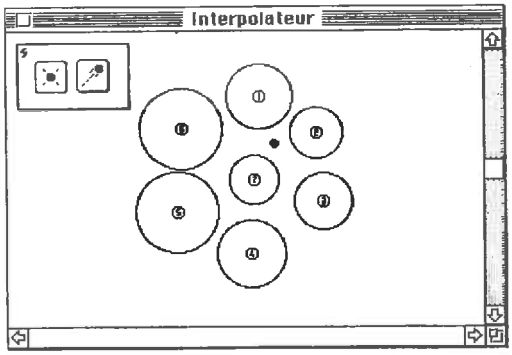


Fig. 5. Des interfaces "à la Syter" permettent de positionner des valeurs de paramètres de synthèse et leur zone d'influence, ici dans des objets Max [19]

On notera qu'ici aucun présupposé n'est fait selon l'usage global ou évolutif des paramètres, mais que celui-ci amènera des stratégies différentes. Si le point représente un aspect global, il faudra disposer d'un autre périphérique pour actionner le son. Typiquement on est dans un geste double ou l'on sélectionne un timbre et on l'on déclenche indépendamment une note. Si le point représente un aspect évolutif, c'est le déplacement du pointeur sur la table qui va créer le son.

3.4 L'espace des phases

La représentation de l'espace sonore peut emprunter des voie diverses, surtout lorsqu'on se concentre sur une faisabilité du son, et non plus simplement une observation du son. Autrement dit, si la préoccupation est l'engendrement d'un son, il faut prévoir un dispositif bijectif entre l'espace sonore et le son lui-même.

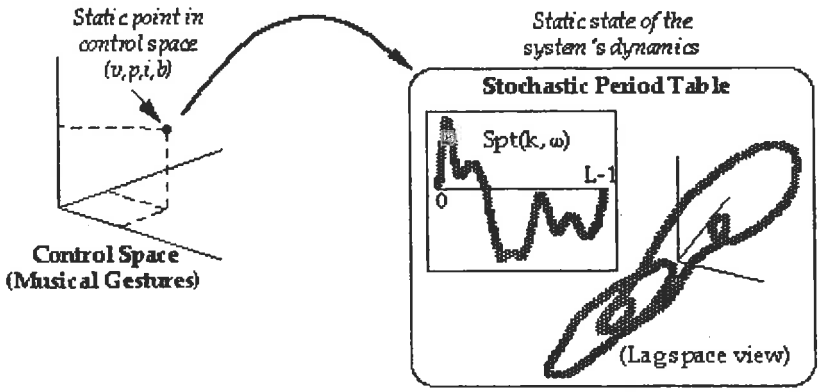


Fig. 6. Des tracés dans l'espace des phases peuvent être mis en relation avec des points de l'espace perceptif pour engendrer des sonorités (Image issue de la thèse de Eric Metois)

Eric Métois a décrit l'usage de l'espace des phases, au sens de la représentation d'un son à l'aide de trois coordonnées: l'échantillon (au sens de valeur instantanée) du signal, et ses deux valeurs précédentes. Le terme phase ici correspond à son usage mathématique au sens fractal du terme et non au sens musical de synchronisation de sinusoïdes. Lorsqu'une trajectoire peut se dessiner, il est alors possible de créer un processus génératif des échantillons successifs. C'est alors l'évolution de la courbe dans cet espace de phases qui va représenter le son. Bien entendu la gouverne de ces courbes est elle-même l'objet d'un espace perceptif. Cet exemple est très instructif pour prouver que les stratégies ne sont pas uniques, et que les plus intéressantes d'entre elles passent par des stratégies combinées.

4. Couplage d'un geste avec un espace perceptif?

Notre description d'un espace perceptif du son va nous aider à formaliser l'espace du geste. En fait la question est: avec ou sans temps? Un geste peut être perçu comme un déplacement dans l'espace d'un ou d'un ensemble de points. Ces gestes peuvent être larges, ou à faible échelle (exemple la pression de la lèvre sur une anche, ou celle du poignet dans la tenue de l'archet), mais la plus grande distinction dans l'intention musicale est : soit on considère que le geste est un, comme un descriptif d'un élément de vocabulaire, soit on considère qu'il est une évolution dans le temps. Prenons l'exemple du langage des mains: dans le langage de signes , on trouvera une signification pour certaines configurations de la main, et pour d'autres un simple passage d'une configuration à une autre. Du côté du geste instrumental , certaines actions sont déclenchantes, et d'autres modulantes: l'appui sur une touche de piano est déclenchant, le souffle de la flûte est modulant. Bien entendu on se trouve souvent dans une configuration mixte, qui est la plus intéressante: déclenchement puis modulation, mais les situations extrêmes existent. La voix en ce sens est exemplaire: le geste est celui de la bouche et des cordes vocales, et l'on trouve à la fois des gestes de décision et des gestes de modulation.

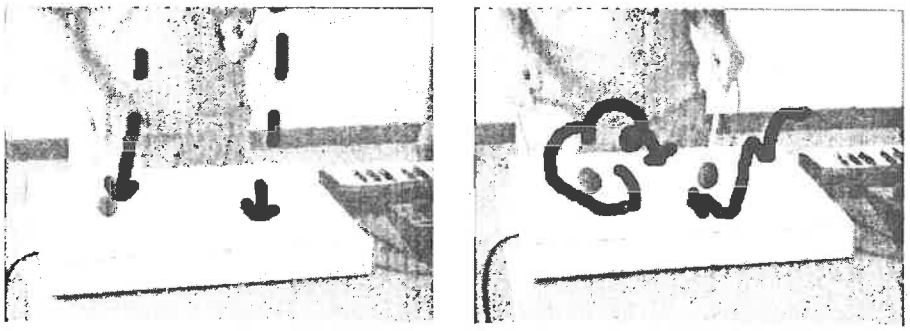


Fig. 7. avec un même périphérique, ici un radio-baton, on peut utiliser des gestes de décision (gauche) ou de modulation (droite) [1]

Cela pose un problème conceptuel pour la représentation du plan intermédiaire dans lequel vont se projeter geste et son. Si l'on est dans une représentation de type modulation, le geste va représenter une trajectoire dans un plan (ou hyperespace) dans lequel la variable temps doit s'intégrer sur la courbe de la trajectoire. Si l'on est dans une représentation de type déclenchement, l'ensemble de la variation temporelle doit être codée dans un point. Dans ce cas le point prend le sens d'un codage symbolique du geste.

5. La réappropriation du geste

L'informatique musicale, dans sa perspective historique, vit un phénomène de balancier : lors de la synthèse hors temps-réel, qui a été conditionnée aussi par l'évolution des machines, le geste physique a disparu (hors de la frappe du clavier numérique) et l'intention musicale s'est exprimée au travers de la composition non seulement de la forme extérieure mais aussi du son lui-même. Des langages comme Music V [12], Csound [6] ou Cmusic ont développé un ensemble de courbes dont le but est de permettre la fabrication et la modification de sons selon un certaine expressivité. Lorsque l'on décide, par exemple dans un acte volontaire de "mise en geste de la musique enregistrée", d'associer aux algorithmes de synthèse un contrôle gestuel, on se réapproprie le geste, puisque la conscience du geste va influer sur l'intention musicale exprimée [1]. Pour être plus clair, l'acte de composition sur ordinateur en synthèse implique de donner toutes les indications de nuance alors que l'acte de mise en geste (gestualisation) reporte certains de ces fonctions dans les données gestuelles. Ce qui est alors apparent est une "corporalisation" des intentions musicales: au lieu d'une intention relatée par une courbe, on a

alors l'apprentissage d'un geste. Il va sans dire que cela peut être pour le meilleur et le pire, mais dans le cas du meilleur, le retour d'écoute est immédiat et engendre dans l'instrumentiste une faculté d'expression indéniable. Max Mathews à l'aide de la "scanned synthesis" [13] démontre parfaitement l'utilisation possible du geste dans des algorithmes fonctionnellement assez simples, comme une corde circulaire vibrant à des fréquences haptiques.

Une autre remarque est que cette réappropriation du geste n'est toutefois pas absolue dans le cas d'une musique numérique dont la commande gestuelle n'est pas elle-même rétroactive. La plupart des instruments de musique acoustiques dérivent d'un couplage entre l'effort que l'on réalise et la teneur du son, ce qui n'est pas nécessairement le cas des capteurs gestuels. Ainsi l'étude de la proprioception, et son espace résultant, reste à intégrer dans un dispositif a retour d'effort, et à coup sûr une bonne adéquation de ce retour d'effort avec l'espace perceptif est nécessaire.

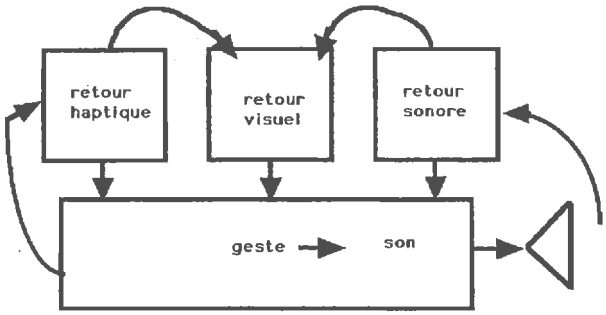


Fig. 8. la fabrication d'un son à l'aide d'un geste dépend étroitement des retours d'effort, d'écoute et du visuel

Même si ce retour proprioceptif (haptique) n'existe pas toujours, le retour par l'écoute est une donnée fondamentale, et certains instruments de musique électronique sont particulièrement bien pourvus de ce côté: citons le thérémine, ou l'instrument de synthèse photosonique. Dans le cas de ce dernier, une précision du geste lumineux d'une fraction de millimètre est aisément atteinte lors de l'écoute des différentes couronnes d'un disque photosonique [4, 5].

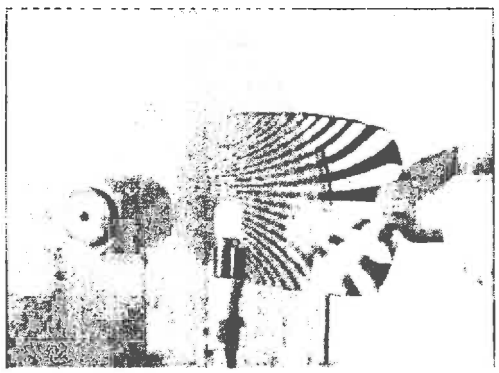


Fig. 9. Dans l'instrument photosonique, la main gauche manipule la position de la lumière tandis que la main droite manipule le filtre optique

D'autre part il ne faut pas négliger les retours par le sens visuel, et l'intermodalité des sens est un domaine de recherche intéressant. Trivialement les indicateurs des niveaux, et autres paramètres de feedback tels les profils de réverbération aident grandement le geste allié à l'écoute [3].

Enfin les capteurs gestuels eux-mêmes [15] ont leurs idiosyncrasies, et le geste, pour pur qu'il soit, dépendra, même pour la musique électronique, d'éléments matériels aux contraintes souvent prégnantes (poids, encombrement, harnachement de certains équipements, etc...)

6. Conclusion

En conclusion, on peut dire que parti d'un propos simple, comment relier son et geste dans un instrument numérique, on a vu que l'évaluation perceptive, et donc aussi le codage symbolique, interviennent dans la définition d'un espace commun, ou partagé entre ces deux domaines. Les notions exposées dans cet article permettent de mesurer la nécessité de définir un cadre conceptuel, notamment pour l'essai de plusieurs périphériques gestuels sur un système de synthèse musicale numérique. Elles sont aussi un élément intéressant dans un domaine connexe qui est celui de la pédagogie musicale, notamment au sujet de la motivation musicale, c'est a dire de l'intérêt d'un instrument.

Références

1. Daniel Arfib & Loic Kessous: "from Music V to creative gesture in computer music", proceedings of the VIIth SBC conference, Curitiba, June 2000, available in CD format (a new version has been accepted for Electronic Musicologic Review)

2. Daniel Arfib: "Digital synthesis of complex spectra by means of multiplication of non-linear distorted sine waves", Journal of the Audio Engineering Society 27: 757-768.

3. Daniel.Arfib: "Visual representations for digital audio effects and their control", proceedings of DAFx99, Trondheim, decembre 1999, pp 63-66

4. Daniel.Arfib, Jacques.Dudon : "A digital version of the photosonic instrument", proceedings ICMC99, Pekin, novembre 1999 , pp 288-290

5. Daniel Arfib & Jacques Dudon: "Photosonic disk: interactions between graphic research and gestural controls", in CD-ROM "Trends in Gestural control of music", editeurs M. Wanderley & M. Battier , publication Ircam, 2000

6. Richard Boulanger: The Csound Book, Perspectives in Software Synthesis, Sound Design, Signal Processing and Programming , MIT press, 2000

7. Gérard Charbonneau: "Timbre and the Perceptual Effects of Three Types of Data Reduction". Computer Music Journal 5(2) p.10-19, 1981.

8. John Chowning: "The Synthesis of Comples Audio spectra by Means of Frequency Modulation. Journal of the audio Engineering Society 21:
561-534. Reprinted in C. Roads and J. Strawn editions 1985 "Foundations of Computer Music". Cambridge, Massachusetts: MIT Press.

9. John Grey: "An Exploration of Musical Timbre". PhD dissertation, department of Psychology, Stanford University, 1975.

et aussi: John Grey: "Multidimensional Perceptual Scaling of Musical Timbres" J.A.S.A vol.61 n°5 May 1977 p 1270-1277

10. Andy Hunt, Marcelo M. Wanderley, Ross Kirk. : "Towards a Model for Instrumental Mapping in Expert Musical Interaction", proceedings ICMC 2000, Berlin

11. Stephen McAdams, Suzanne Winsberg, Sophie Donnadieu, Geert De Soete, Jochen Krimphoff; "Perceptual scaling of synthesized musical timbres: common dimensions, specificities, and latent subject classes", Psychological Research, 58, 177-192 (1995)
disponible sur <http://mediatheque.ircam.fr/articles/textes/McAdams95a/>

12. Max Mathews: "The Technology of Computer Music" (1969). MIT Press, Cambridge, MA.

13. William Verplank, Max V. Mathews, Robert Shaw: "Scanned Synthesis, proceedings of the Icmc 2000, Berlin

14..Eric Metois: "Musical Gestures and Audio Effects Processing", DAFx98 conference, barcelona,
disponible sur <http://www.iua.upf.es/dafx98/papers/>

15. Joe Paradiso: "American ionnovation in electronic musical instruments",
disponible sur <http://www.newmusicbox.org/third-person/index_oct99.html>

16. Jean-Claude Risset et Wessel: "Exploration of Timbre by Analysis and Synthesis", The Psychology of Music, Deutsch eds. Orlando Academics, 1982.

17. Sylviane Sapir: "Interactive digital audio environments: gesture as a musical parameter", proceedings of DAFx00,
disponible sur <http://profs.sci.univr.it/~dafx/papers.html>

18. Wessel David L.: "Timbre Space as a Musical Control Structure", Rapport Ircam 12/78, 1978,
disponible sur <http://mediatheque.ircam.fr/articles/textes/Wessel78a/>

19. Vect pour Max Macintosh, vectFat 1.1 (par Adrien Lefevre),
disponible sur <http://www.adlef.com/soft/vect/vectdoc.html>

20. A. Slawson "Vowel quality and musical timbre as function of spectrum envelope and fundamental frequency" 1968,Journal of Acoustic Society of America, n°43, p 87-101.

21. Thierry Rochebois, Thèse, disponible sur <http://www.ief.u-psud.fr/~thierry/these/>

22. Claude Cadoz & Marcelo M. Wanderley: Gesture-Music, in M. Wanderley and M. Battier (eds): Trends in Gestural Control of Music- Ircam - Centre Pompidou, 2000.

Jongleries musicales

Aymeric Willier
UTC UMR 6600
BP 20529
60205 Compiègne Cedex France
aymeric.willier@utc.fr

Résumé : Le jonglage est un art qui s'adresse essentiellement à notre vision. Cependant, de plus en plus de spectacles, résultants d'un travail conjoint de jongleurs et de musiciens, révèlent l'intérêt de celui-ci pour notre ouïe. Le travail présenté ici a pour objectif d'offrir au jonglage une interaction encore plus directe avec le sonore : il propose de recycler les gestes effectués par le jongleurs pour mouvoir les objets, en gestes de contrôle de la musique. L'interface présentée repose sur le traitement des signaux électromyographiques des principaux muscles utilisés pour le jonglage.

Mots clefs : Jonglage, Musique , interaction, Interface Midi , Electromyographie.

1 Jonglage et musique : des bases pour une interaction

1.1 Motivations

Le jonglage est d'évidence un art qui sollicite chez nous essentiellement un sens : la vue. Dés lors la musique, qui elle, s'adresse principalement à l'ouïe, est souvent exploitée conjointement au jonglage. De façon très simple, elle peut être une ponctuation des temps forts, comme elle l'est dans l'utilisation du roulement de tambour dans un numéro de jonglage basé sur l'exploit. Lorsque le numéro de jonglage se théâtralise, la musique vient compléter les personnages, les ambiances, ou les sentiments. Dans cette dernière approche, l'accompagnement d'une bande sonore, celle d'un petit orchestre, remplit tout à fait les attentes. Qui plus est, la relation entre jonglage et musique n'est pas directe : elle passe par le sentiment, le personnage, l'ambiance qui tient de la théâtralisation, de la mise en scène du spectacle : il s'agit plus ici de mise en relation simultanée du jonglage et de la musique au travers de la mise en scène, que d'une mise en relation directe du jonglage et de la musique.

Une relation plus directe entre jonglage et musique naît avec l'exploitation des « bruits de jonglage ». Les bruits de jonglage sont ceux que peuvent produire mécaniquement les objets manipulés par le jongleur, eux-mêmes, ou au contact d'autres objets. Furent-ils jusqu'à présent considérés comme indésirables, et alors couverts si possible par la musique du numéro, ou ignorés justement, car la musique les étouffait, en empêchant leur perception par le spectateur, voire par le jongleur lui-même ? Quoiqu'il en soit, les jongleurs les ont progressivement admis comme partie intégrante de leur prestation, au point d'en travailler la production. L'un des principaux intérêts de l'utilisation des bruits de la jongle est la relation de cause à effet entre les gestes du jongleurs et leur production. Leur limitation principale se révèle rapidement dans les sons qu'il est possible de générer. Ceux–ci restent tout d'abord ceux physiquement productibles par les balles, massues et autres objets jonglés qui ne sont pas conçus à des fins musicales.

Deux réactions à cette limitation ont alors été constatées. La première est la transformation d'objets usuels du jonglage dans le but d'en accroître le potentiel sonore : exploitation des vibrations d'une ficelle de diabolo, à l'identique de celle des vibrations d'une

corde de guitare, adjonction aux massues de grelots ou de hochets, ou fixation de sonnailles sur les cerceaux, balles remplies de grelots (1) ... De même est exploitée la rotation du diabolo comme excitation de résonateurs acoustiques fixés sur ce dernier...marquant ainsi un retour aux origines de cet ancien leurre guerrier. Les bruits d'impact de balles de rebonds, sur des planchers quelconques, puis sur des « cubes de rebond », cavités résonnantes pouvant être accordées, ont également été exploités (2). La seconde réaction est de jongler des objets moins usuellement jonglés, afin d'exploiter leur potentiel sonore, comme par exemple des barres métalliques creuses, à la place des massues sonnant sourdement à chaque réception dans la main (3).

A l'observation de l'évolution de l'utilisation des bruits de rebond de balles, on constate qu'après la découverte d'un potentiel musical, les efforts sont faits pour accroître et maîtriser ce potentiel autant que le permet la réalité physique des objets manipulés. Ainsi l'impact de rebond génère d'abord un son indésirable, puis un son désiré et utilisé, jusqu'à un son « élaboré » via l'accord de la cavité sur laquelle est pratiquée le rebond. Ce dernier point est important. Il marque la volonté d'accéder à une possible sélection du son, sélection qui reste cependant limitée : on ne jonglera jamais avec des cloches de trois tonnes, même si d'entendre une cloche sonner à chaque réception ou rebond de balle pourrait servir le spectacle. On peut aussi regretter une autre limite importante : en effet si on peut espérer obtenir certains sons d'une massue maracas, les sons que l'on obtiendra ne seront que ceux que les massues peuvent produire dans des mouvements de jonglage exclusivement. Pour obtenir d'autres sons il faudra modifier le mouvement de jonglage, dans la limite du possible, et le maximum de variation possible pour un mouvement ne permet trop souvent pas la production du son espéré.

1.2 Nouvelles possibilités

Pour passer outre ces dernières limites évoquées, il faut rendre possible la production d'autres bruits, d'autres sons, qui soient directement déclenchés par le jonglage, comme s'ils étaient intrinsèquement le résultat du jonglage, mais choisis pour leur caractéristiques propres, et non pas imposés par la réalité physique de l'objet manipulé et des possibles dans la manipulation. Néanmoins, préserver la relation de cause à effet, appréciée dans les bruits de jonglage, est primordial. Or, le principe sur lequel se construit cette relation consiste en l'utilisation pour la génération de sons, des mouvements que le jongleur utilise pour mettre les objets manipulés en mouvement, ou pour les garder immobiles. A la fonction de manipulation d'objet, de mise en mouvement d'objets, on doit donc adjoindre une fonction de génération sonore. Le mouvement de jonglerie et l'énergie qu'il nécessite, peuvent servir comme déclencheur et contrôleur de sons. Les technologies actuelles permettent d'extraire de gestes et de mouvements des informations sur ceux-ci, via des dispositifs de captation de mouvement. L'information obtenue par ces dispositifs pourra être utilisée pour la commande musicale (4).

1.3 Captation de mouvement et musique

La captation de mouvement compte de nombreux domaines d'application : biomécanique, télésurveillance, robotique, sécurité,... Côté pratiques artistiques, ce sont les danseurs qui ont le plus exploré l'idée de lier mouvement et musique (5)(6)(7)(8). Ils ont donc imaginé et réalisé à cet effet un grand nombre de dispositifs de captation de mouvement. Les travaux très exhaustifs de A.Mulder (9) et D.Roger (10), respectivement sur les dispositifs de captation de mouvement en général et sur ceux dévolus à la commande musicale en

particulier, proposent une revue complète des technologies utilisées. Les systèmes de captation de mouvement sont donc très nombreux, d'autant plus que chaque artiste ayant exprimé le besoin d'un tel dispositif, a pu proposer une solution technique adaptée à son problème. De plus, les informations obtenues par captation de mouvement une fois encodées en MIDI, les possibilités ne se limitent plus à la commande musicale : le langage Midi utilisé pour relier les instruments de musique électronique, est aisément traduisible en langage de commande pour tout autre dispositif, qu'il s'agisse d'éclairages, d'artifices, de vidéo, de machineries... Apparaissent alors des possibilités de mise en rapport direct de nombreux éléments, traités souvent isolément dans un spectacle de jonglage : il ne reste qu'à les exploiter.

2 Proposition d'interface

2.1 Principe

Les principaux systèmes de captation de mouvement existants ont une approche cinématique du mouvement : évaluation des déplacements, accélérations, vitesses... Le dispositif présenté ici se base sur une approche dynamique du mouvement : ce n'est pas le mouvement dans sa réalisation qui est évalué, mais les efforts à l'origine de ce mouvement. Ces derniers peuvent être évalués par l'enregistrement de l'Electromyogramme, ou EMG. L'EMG est l'enregistrement d'un signal électrique capté à la surface des muscles, résultant de la mise en activité des différentes cellules musculaires. L'origine de ce signal est telle qu'il est qualitativement lié aux forces produites par le muscle (11). Son exploitation en commande musicale n'est pas nouvelle, puisque les professeurs H.Lusted et B.Knapp (11) (12) de l'Université de Stanford, ont mis en œuvre un dispositif appelé *Biomuse*, utilisé actuellement par Atau Tanaka pour ses prestations scéniques personnelles (13), ainsi que celles qu'il assure au sein du Sensorband (14). Une autre utilisation de l'EMG liée à la musique a été proposée au MIT par T.Marrin (15). T.Marrin enregistre l'EMG ainsi que d'autres signaux physiologiques du chef d'orchestre, et exploite les relations entre ces signaux et les intentions musicales que le chef d'orchestre veut communiquer aux musiciens. Notre démarche diffère de ces deux approches : nous ne proposons pas un instrument pour lequel il faut se construire un ensemble de mouvements et gestes maîtrisés dévolus à son utilisation, comme c'est le cas pour le *Biomuse* ; nous ne nous intéressons pas non plus à des mouvements et gestes effectués dans un cadre musical comme c'est le cas pour T.Marrin ; notre démarche est de recycler des gestes experts effectués dans le cadre d'un art choisi, ici le jonglage, en geste de commande musicale.

2.2 Choix de l'EMG

La première motivation d'utilisation de l'EMG vient de ce que l'équipement de recueil de ce signal n'obstrue absolument pas le mouvement. L'articulation à laquelle on s'intéresse reste totalement libre, puisque les muscles qui en assurent le mouvement sont localisés en amont : pour exemple, dans le cas du poignet, les fléchisseurs et extenseur de cette articulation sont situés à proximité du coude sur l'avant bras. Ceci est très important dans le cadre du jonglage : le dispositif ne doit en effet pas encombrer les articulations des membres supérieurs.
Une seconde motivation repose sur la constatation suivante : la production d'un mouvement maîtrisé sollicite les muscles à son origine dans une chronologie invariante et optimisée. Cette chronologie s'affine et s'élabore avec l'apprentissage du mouvement (16). Ceci se révèlera

particulièrement vrai pour le jonglage, et nous permettra une localisation temporelle précise des phases du mouvement. Cette chronologie optimisée n'empêche pas une certaine variation dans l'amplitude et le contenu du signal EMG, variation dont l'exploitation est souhaitable musicalement. L'EMG, signal physiologique, présente une richesse qui reste encore à interpréter.

2.3 Mise en œuvre du dispositif

La mise en œuvre de ce dispositif passe par diverses étapes. La première est l'analyse du geste considéré et le choix des muscles à observer. La seconde consiste en l'enregistrement de signaux EMG durant le jonglage pour finalement en proposer les traitements et le conditionnement Midi de l'information extraite de ces signaux.

2.3.1 Analyse du geste considéré :

Dans un premier temps, nous nous sommes intéressés au jonglage de balles, nous limitant même au jonglage à trois balles. Le mouvement de base dénommé cascade, consiste en des lancers alternés de chaque main. Aussi primaire soit elle, la cascade présente deux des huit actions élémentaires par lesquels Durant et Pavelack (17) proposent de décrire tout mouvement de jonglage : le lancer et l'amorti.

Durant la Cascade, on observe un mouvement de circumduction de l'avant bras. Par ailleurs, le rôle important de la main dans le lancer et l'amorti de la balle suggère une activité de maintien au niveau du poignet. Ces deux constatations nous ont guidé dans le choix des muscles à observer :

- muscles assurant cette circumduction, qui peut être décrite comme une composition de flexion et d'extension alternées de l'avant bras, en quadrature de phase avec des rotations externe et interne alternées du bras.
- muscles assurant le maintien du poignet, soit les fléchisseurs et extenseurs de celui-ci.

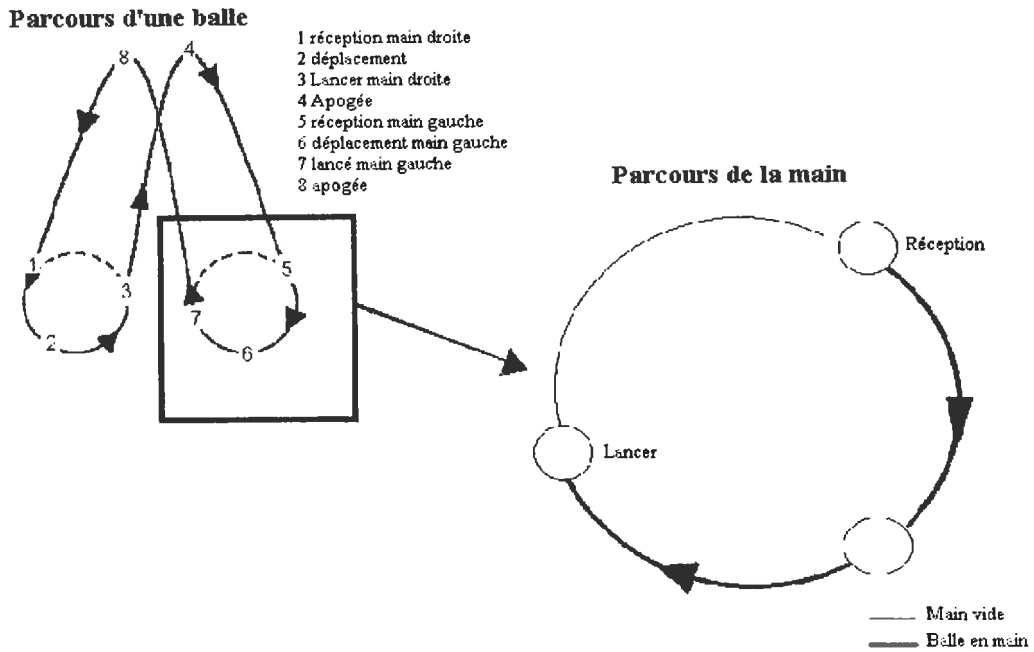


Figure 1 : Cascade à trois balles : Trajectoire de la balle et déplacement d'une main

Notre dispositif reposant sur le recueil de l'EMG de surface, notre choix de muscle s'est vu limité aux muscles affleurant directement sous la peau. Les muscles choisis sont donc :

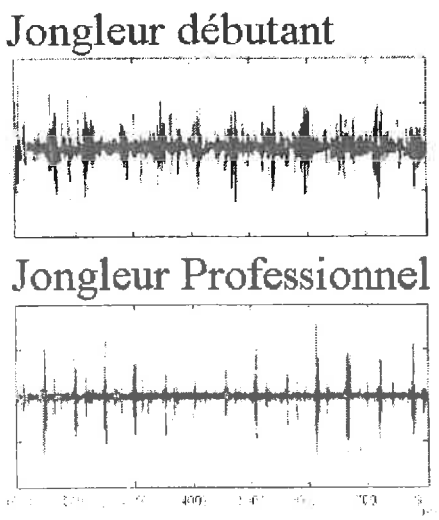
Palmaris Longus (PL): grand palmaire fléchisseur du poignet
Extensor Carpi Radialis Brevis (RB): extenseur court radial de la main, extenseur du poignet
Biceps Brachii (BB) : biceps, fléchisseur du coude
Triceps Brachii (TB): triceps, extenseur du coude
Pectoralis Major (PM): grand pectoral, rotateur interne du bras
Teres minor (TR): petit rond, rotateur externe du bras

L'équipement matériel nous limitant à l'observation simultanée de 4 EMG, nous nous sommes, dans un premier temps, intéressés à l'observation des PL, RB, BB et TB, ou de deux de ces muscles pour les deux bras simultanément.

2.3.2 Enregistrements

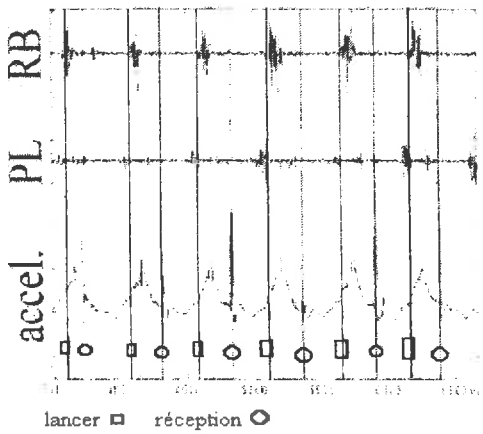
Nous avons pu effectuer des enregistrements des EMG de TB, BB, RB et PL chez des jongleurs ayant une pratique amateur ou professionnelle. La comparaison des enregistrements effectués chez les uns et les autres confirme la clarification de la chronologie des EMG avec l'apprentissage du mouvement (Figure 2), confirmant nos motivations pour l'utilisation de l'EMG (Cf. 2.2). Une première observation de l'ensemble des signaux nous a invité à nous focaliser sur l'observation des RB et PL, révélant une activité ponctuelle chronologiquement corrélée aux mouvements du jonglage. Des enregistrements supplémentaires avec utilisation simultanée de divers capteurs nous ont permis de localiser les activités ponctuelles du RB au moment du lancer (Figure 3). La détection de ces activités permettrait donc aisément la synchronisation d'un événement sonore et d'un lancer.

Figure 2 : effet de l'entrainement



EMG du PL chez un jongleur amateur débutant et chez un jongleur professionnel lors de la cascade à trois balles.

Figure 3 :



Deux dispositifs, faisant pour l'un appel à un accéléromètre, pour l'autre à un capteur à effet Hall et à un aimant placé dans une balle nous ont permis de localiser la réception et le lancer de la balle relativement aux bouffées d'activité des RB et PL.

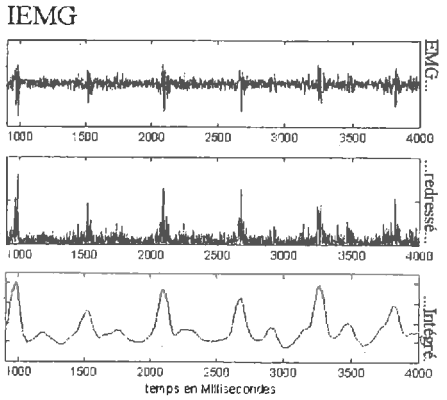
2.3.3 Traitements proposés

Deux traitements avant conditionnement Midi se sont imposés : l'EMG intégré ou IEMG, et la détection de pic d'activité.

L'IEMG nous donne une information continue, estimation de l'énergie du signal. L'IEMG peut être facilement utilisé comme contrôle continu. Nous le conditionnerons en Midi comme un contrôleur continu. Atau Tanaka en suggère l'utilisation pour le contrôle de l'évolution de paramètres de timbre (18). Son calcul est différent s'il est évalué en analogique ou en numérique. Ayant à l'utiliser dans ces deux cas, nous effectuons en fait un redressement d'une moyenne quadratique (RMS) sur une période fixe, en analogique, et un redressement suivi d'une moyenne des N derniers échantillons, en numérique (19) (Figure 4).

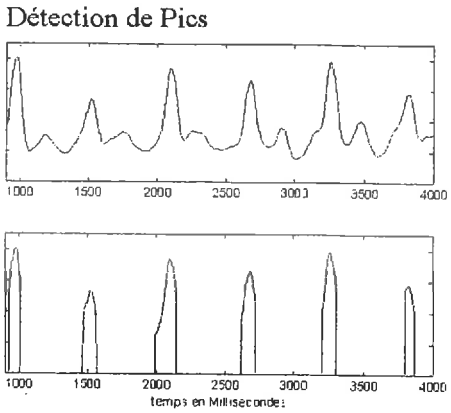
La détection de pic effectuée sur cet IEMG nous donne une information discrète, qui dans le cas du RB, nous permettra de synchroniser un événement sonore au lancer. Nous le conditionnerons d'ailleurs en Midi comme note on/off. Nous l'effectuons actuellement sur l'IEMG évalué analogiquement. Nos premiers essais se basant sur un seuil fixe ont révélé une grande variabilité dans les résultats, les pics d'activité montrant, selon les jongleurs, une grande variation dans leur amplitude. Ceci nous a poussé à mettre au point un principe de seuillage avec seuil actualisé dynamiquement, plus performant pour l'ensemble des jongleurs, mais plus complexe à mettre en œuvre en temps réel (Figure 5).

Figure 4



IEMG :
Les étapes du traitement de l'EMG pour obtention de l'IEMG sont présentées ici pour un signal recueilli sur le RB.

Figure 5



Détection de Pics : Résultats de la détection de pics avec actualisation dynamique de seuil pour ce même signal.

2.4 Configuration matériel

Les EMG sont captés par des électrodes posées par paire à la surface des muscles considérés. La pose des électrodes nécessite la préparation de la peau afin d'en réduire la résistance électrique. Nous utilisons une système de télémétrie réalisé par John+Reilhofer assurant l'acquisition et la transmission HF de quatre EMG.

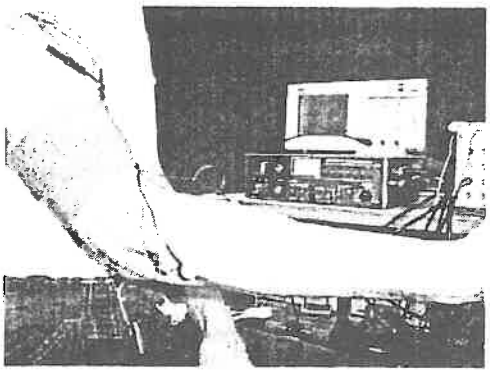


Figure 6 : Electrodes posées sur le RB. Le récepteur du système de télémétrie est partiellement visible à droite.

Le dispositif actuellement réalisé procède ensuite au calcul analogique de l'IEMG à partir des EMG. Ces IEMG font ensuite l'objet d'une conversion numérique à une fréquence d'échantillonnage de 200Hz. Nous effectuons le conditionnement des signaux IEMG en un contrôleur continu Midi pour chacun des quatre IEMG. Lors de la détection d'un pic d'activité, nous envoyons un signal « note on » sur une note choisie pour chaque EMG considéré.

Un dispositif plus performant est en cours de réalisation. Les EMG font directement l'objet d'une conversion analogique/numérique à une fréquence d'échantillonnage de 2000 Hz. Le calcul numérique des IEMG, leur conditionnement en Midi, ainsi que la détection des pics d'activité, via un principe de seuillage dynamique, font l'objet d'une programmation sous HpVee.

3 Conclusion et perspectives

Nos travaux nous ont donc amenés à une analyse des phases des mouvements durant le jonglage de balle, au recueil d'EMG de muscles choisis en fonction de cette dernière analyse, et à proposer un premier dispositif. Ce dernier procède au calcul de l'IEMG, et à son conditionnement comme contrôleur Midi continu, ainsi qu'à la détection de pics sur ces IEMG, dont le résultat est conditionné en signal note on/off, permettant en particulier la synchronisation d'événements sonores avec les lancers de balles.

La réalisation de ce dispositif trouve sa motivation dans l'exploration des interactions possibles entre musique et jonglage. Les choix technologiques, nos options de travail, réduisent cependant la portée de nos résultats : le jonglage de massue ne sollicitera assurément pas les muscles choisis ici de la même façon que le jonglage de balle, d'autres muscles s'avèreront peut être alors plus intéressants à l'observation. Nos premières observations des muscles rotateurs externe et interne du bras sont dans ce sens prometteuses. Par ailleurs, nous explorerons prochainement à l'occasion d'une semaine de travail au STEIM, l'utilisation d'autres capteurs dans le cadre d'autres disciplines de jonglage telles que le swinging, le diabolo, ou le rebond de balle.

D'un point de vu matériel, le dispositif en est actuellement à sa première réalisation, dont les limitations sont évidentes. Nos compétences nous ont invité à utiliser, dans un premier temps, des outils peu adaptés, mais la conception d'un dispositif complet des capteurs au conditionnement Midi apparaît réalisable. C'est l'objectif fixé pour le dispositif en projet. L'utilisation de convertisseur signal analogique/Midi comme l'Atomic (20), le Sensorlab (22) ou encore l'I-cube (21) est également envisageable.

Du point de vue des traitements possibles de l'EMG, nous nous tournons actuellement vers les outils d'analyse temps fréquence (23) avec l'espoir d'extraire de l'EMG des informations sur la dynamique des lancers. A partir de ces informations, nous espérons classer les lancers en fonction de leur hauteur : l'historique des lancers et de leur hauteur peut

permettre de déduire la figure en cours de réalisation. Il serait alors possible d'affecter à chaque figure des paramètres musicaux propres, donnant ainsi au jongleur un moyen supplémentaire de sélection de sa musique.

Remerciements

L'auteur tient à remercier Catherine Marque, Frédéric Pradal et Hélène Willier pour leurs suggestions et commentaires.

Références

(1) Chant de balles, Vincent De Lavenere http://www.chantdeballes.com/contact.html
(2) Kabbal http://www.lefourneau.com/creations/00/kabbal/kabbal.html
(3) Jörg Müller http://perso.infonie.fr/ezec/Ezec/ebase.htm
(4) T. Winkler, ``Making motion musical: Gestural mapping strategies for interactive computer music.'' in Proc. Int. Computer Music Conf. (ICMC'95). 261-264. (1995)
(5) Siegel, W. "The Challenges of Interactive Dance - an overview and case study." Computer Music Journal 22(4). http://www.daimi.au.dk/~diem/dance.html (1998)
(6) T. Winkler, ``Motion-sensing music: Artistic and technical challenges in two works for dance.'' in Proc. Int. Computer Music Conf. (ICMC'98). 471-474. (1998)
(7) Dancetech http://art.net/~dtz/
(8) Bodysynth http://www.synthzone.com/bsynth.html
(9) Mulder, A. Human movement tracking technology, Tech. Rep., Simon Fraser University, school of kinesiology. (1994)
(10) Roger, D. Motion Capture in Music: Research http://farben.latrobe.edu.au/motion/ (1998)
(11) DeLuca, C. J. "The use of Electromyography in biomechanics." J. Applied Biomechanics 13: 135-163. (1997).
(11) Lusted, H. S. and B. Knapp. Controlling computers with neural signals. Scientific American. http://www.sciam.com/1096issue/1096lusted.html (1996)
(12) Putnam, B. The use of the electromyogram for the control of musical performance. Ph.D. Dissertation, Stanford University. (1993)
(13) A. Tanaka, ``Musical Technical Issues in Using Interactive Instrument Technology with Application to the Biomuse,'' in Proc. Int. Computer Music Conf. (ICMC'93). 124-126. (1993).
(14) Bongers, B. "An interview with Sensorband." Computer Music Journal 22(1): 13-24. (1998)
(15) Marrin, T. and R. Picard. A methodology for mapping gestures to music using physiological signals. Presented at the International Computer Music Conference, Ann Arbor, Michigan. (1998)
(16) Brénière Y. , L. Gatti et M.C. DO Description biomécanique de l'arraché à deux bras, présenté au Cogress of the society of Biomechanics. Lille. (1979)
(17) Durand, F. and T. Pavelack. Le psychojonglage, le livre de la jongle_. Toulouse. (1999)
(18) Tanaka, A. Musical performance practice based on sensor-based instruments. Trends in gestural Control of Music. I.-C. Pompidou. Paris. IRCAM-Centre Pompidou. (2000)
(19) Duchêne, J. and F. Goubel "Surface Electromyogram during voluntary contraction : Processing Tools and Relation to physiological events." Critical Reviews in Biomed. Eng. 21(4): 313-397. (1993)
(20) http://www.ircam.fr/produits/technologies/atomic/Atomic.html
(21) http://www.infusionsystems.com/
(22) http://www.steim.nl/products.html
(23) Karlson , S. J. YU and M. Akay . Time-Frequency analysis of myoelectric signals during dynamic contractions: A comparative study. IEEE Transaction on Biomedical Engineering. 47(2): 228-237.(2000)

CliMAX : environnement de création musicale pour enfants

Murray Frédéric (étudiant à la maîtrise)
Faculté de Musique, Université Laval
Québec, Canada, G1K 7P4
fm@fmurray.com

En hommage posthume à Louis Daignault, décédé le 23 mai 2000

Résumé :
CliMAX, environnement de création musicale conçu à partir du logiciel MAX, vise à rendre la programmation musicale accessible aux enfants. Ce logiciel utilise des objets « préfabriqués » qui cachent les processus complexes de MAX et comporte trois types d'objets : les objets d'entrée, les objets de sortie et les objets de traitement. Les compositions musicales de trois enfants sont examinées et classées parmi les trois types d'approches quant à l'utilisation de l'environnement par les jeunes : l'approche instrumentale, l'approche par séquence et l'approche exploratoire. CliMAX se différencie d'un logiciel séquenceur par son concept visuel entrée-traitement-sortie. Finalement, quelques avenues possibles concernant l'avenir de CliMAX sont examinées. CliMAX est disponible gratuitement sur le site du chercheur : www.fmurray.com

Mots clefs : aide à la composition musicale, composition musicale chez les enfants, création musicale chez les enfants, langage de programmation musicale, MAX.

Contexte

L'utilisation d'un langage de programmation pour la composition est une avenue de recherche pratiquement inexplorée en éducation musicale (Holland 1989). En effet, la recherche sur les applications pédagogiques de l'ordinateur en musique s'est concentrée presque exclusivement sur le développement de didacticiels (ex. exerciseur en formation auditive) et, dans une moindre mesure, sur l'utilisation de logiciels-outils tels que les éditeurs de partitions musicales (Higgins 1992). L'apparition d'un nouveau langage de programmation musicale MAX (version 3.5) potentiellement accessible aux jeunes, laisse toutefois présager de nouvelles avenues de recherche.

CliMAX est donc né d'une recherche antérieure qui avait pour principal objectif de rendre la composition musicale avec MAX accessible aux jeunes. Pour ce faire, une mise en parallèle des processus de programmation musicale et de la qualité des produits artistiques qui en résultent a été effectuée. Les résultats de cette recherche sont présentés dans un article antérieur (Daignault et Murray, 2001).

Le présent article décrit, d'une part, l'environnement de création musicale CliMAX[1] et, d'autre part, présente trois exemples de créations musicales de jeunes[2].

---

[1] CliMAX est l'acronyme de Composition Ludique Interactive avec MAX. Le terme ludique souligne la ressemblance entre MAX et un jeu de construction de type « Lego » où il est possible d'assembler des blocs pour construire des structures particulières. Le terme interactif fait référence à l'interactivité que l'on retrouve entre le jeune et l'ordinateur dans notre environnement.

[2] Les jeunes de l'étude principale étaient âgés de 12 et 13 ans. Cependant, selon notre expérience de CliMAX et l'expérience musicale des jeunes, cet âge peut varier plus ou moins.

Développement de l'environnement de création musicale

Programmer avec MAX consiste à relier des objets visuels à l'écran de manière à réaliser différentes structures musicales[3]. La première étape de l'étude initiale (Daignault et Murray, 2001) consistait à choisir les objets de MAX appelés à être utilisés par les jeunes. Plusieurs objets de MAX sont très complexes et ont été éliminés dès le départ. De plus, nous avons constaté que les combinaisons possibles des objets conservés devenaient à leur tour très complexes pour les jeunes. Par exemple, pour créer une structure musicale permettant de modifier les timbres, les durées, les hauteurs et les intensités des notes, il faut posséder une connaissance approfondie de la programmation avec MAX que n'ont évidemment pas, a priori, les enfants visés par ce type d'environnement.

À partir de ce constat, nous avons décidé d'utiliser une technique fréquemment utilisée avec le logiciel MAX. Cette technique, appelée « encapsulation », permet de créer un nouvel objet (nouvelle boîte à l'écran) qui cache un mécanisme complexe. La figure 1 représente le résultat de la technique de l'encapsulation pour l'objet « délai ».

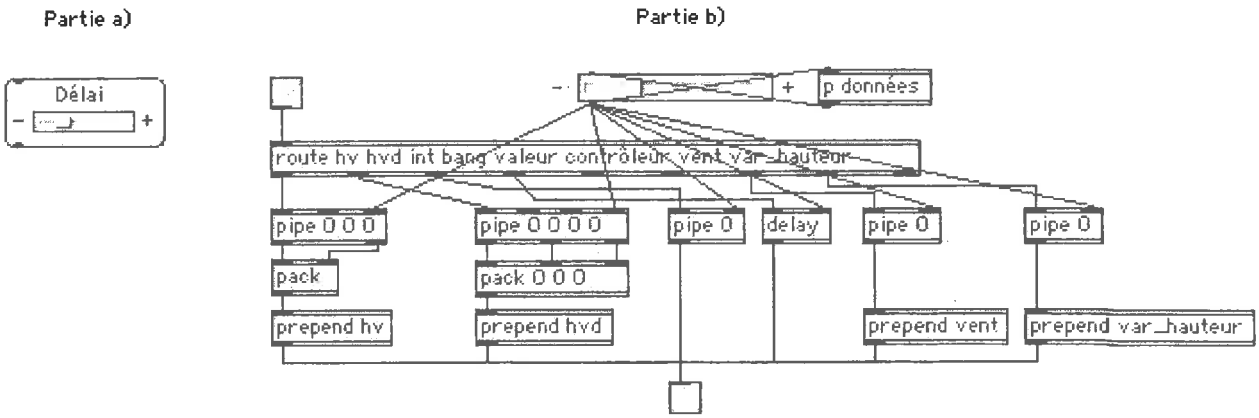


Figure 1. Utilisation de l'encapsulation pour simplifier l'objet délai.

Dans la partie a) de cette figure, nous distinguons l'objet délai qui, comme son nom l'indique, permet d'obtenir un délai de la ligne musicale (effet d'écho). Dans la partie b), nous retrouvons la structure musicale complète de l'objet délai, son fonctionnement interne, en fait, qui est rendu invisible grâce à l'encapsulation. La partie a) de la figure 1 est donc en tout point équivalente à la partie b).

CliMAX est donc constitué d'un ensemble d'objets « préfabriqués » qui camouflent les éléments habituellement plus complexes de MAX. Nous avons construit les objets préfabriqués en voulant conserver une simplicité et une cohérence pour l'ensemble de ces nouveaux objets et en respectant les concepts d' « inlet » et d' « outlet » de MAX[4].

En définitive, CliMAX désigne l'ensemble des objets préfabriqués conçus avec MAX qui constitue un environnement de composition musicale pour les jeunes. Dans CliMAX, les jeunes déplacent à l'écran des objets qui représentent des concepts de composition musicale (délai, transposition, miroir, etc.) et les relient entre eux, par des connexions, pour produire des structures musicales plus ou moins complexes dans le but de créer une composition

[3] Pour un historique de MAX et une discussion sur la place de celui-ci dans les langages de programmation, on se réfèrera à l'article de François Déchelle dans la bibliographie.

[4] Dans MAX, chaque objet, à quelques exceptions près, possède une ou plusieurs entrées (*inlet*) pour recevoir des messages et une ou plusieurs sorties (*outlet*) pour envoyer des messages.

originale. Plusieurs objets de CliMAX communiquent avec des périphériques MIDI notamment un clavier MIDI, une flûte MIDI et un module de son[5].

L'espace de travail de CliMAX, représenté par la figure 2, se divise en quatre parties : les objets d'entrée, les objets de sortie, les objets de traitement et l'espace de construction.

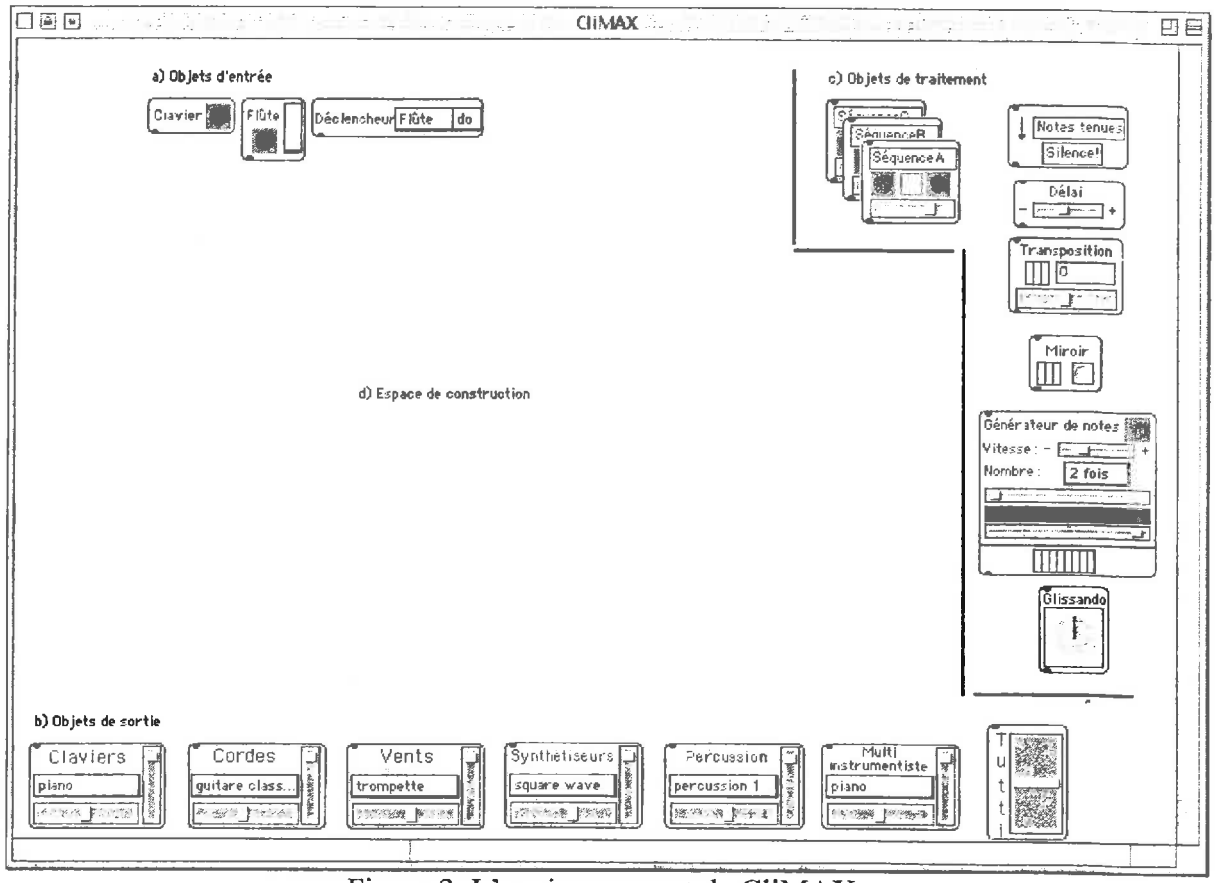


Figure 2. L'environnement de CliMAX.

1. Les objets d'entrée

Les objets d'entrée (lettre a de la figure 2) **reçoivent des données** (hauteur, intensité et durée des sons) provenant des instruments MIDI qui sont branchés à l'ordinateur. Ces objets sont colorés en bleu dans CliMAX et se retrouvent dans le coin gauche supérieur de l'écran.

2. Les objets de sortie

Les objets de sortie (lettre b de la figure 2) permettent la production des sons dans CliMAX. Ces objets **envoient les données** adéquates au module de son (branché à l'ordinateur) qui réagit en produisant les sons nécessaires. Les objets de sortie se divisent en plusieurs familles d'instruments (claviers, cordes, vents, synthétiseurs, percussion et multi-instrumentiste). Ces objets sont colorés en vert dans CliMAX et se retrouvent au bas de l'écran.

[5] Il est possible de créer de nouveaux objets pour communiquer avec d'autres périphériques MIDI.

3. Les objets de traitement

Les objets de traitement (lettre c de la figure 2), **font subir un traitement précis** aux informations musicales qu'ils reçoivent. Ainsi, l'objet transposition, par exemple, transpose une ligne mélodique d'une valeur déterminée par l'usager. Dans CliMAX, les objets de traitement sont colorés en jaune et se retrouvent dans le côté droit de l'écran.

4. L'espace de construction

L'espace de construction (lettre d de la figure 2) est une partie de l'écran où on assemble les objets pour créer une structure musicale. Pour une description complète des fonctions de tous les objets de CliMAX, se référer à l'appendice A.

Exemples de créations musicales

Examinons maintenant trois structures musicales construites par des jeunes avec CliMAX[6]. Dans l'étude principale (Daignault et Murray, 2001), les jeunes disposaient de trente minutes pour préparer leur composition; bien entendu, chacun d'eux avait pu se familiariser au préalable avec CliMAX.

La composition de François

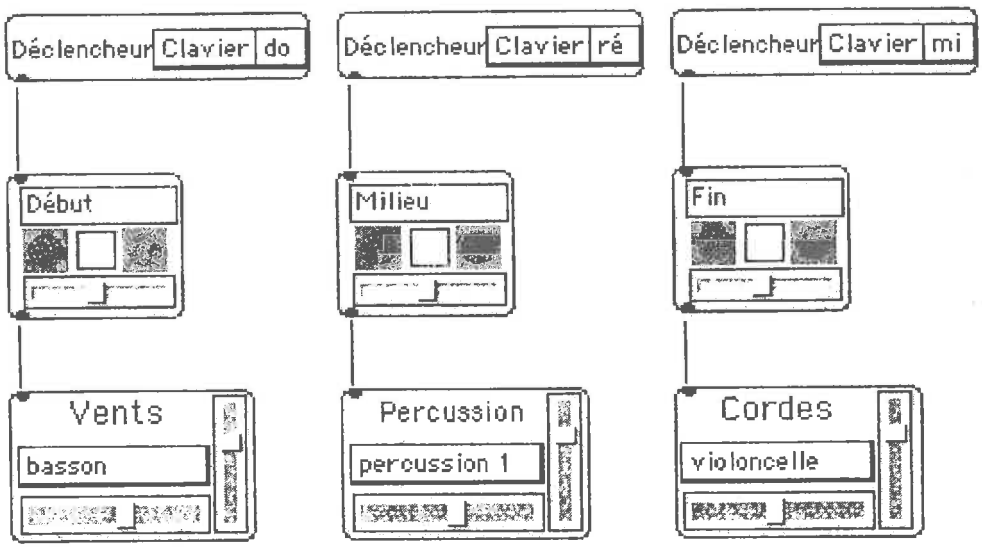


Figure 3. La composition de François.

François utilise trois objets « séquence » dans sa composition. Ces objets, nommés « Début », « Milieu » et « Fin » par l'enfant, peuvent être comparés à des mini-enregistreuses. L'enfant avait d'abord employé l'objet clavier (qui reçoit des données du clavier MIDI) et avait relié celui-ci aux objets séquence pour enregistrer une mélodie. Puis, au moment de faire jouer sa composition, il utilise trois objets « déclencheur » de façon à démarrer chacun des objets séquence. Ainsi, François joue au chef d'orchestre en utilisant les notes *do*, *ré* et *mi* du clavier MIDI.

Finalement, remarquons que François emprunte des sons de percussion pour la séquence identifiée « Milieu ».

[6] Les compositions musicales des trois jeunes peuvent être entendues dans la section « Technologie musicale » du site : www.fmurray.com

La composition de Tristan

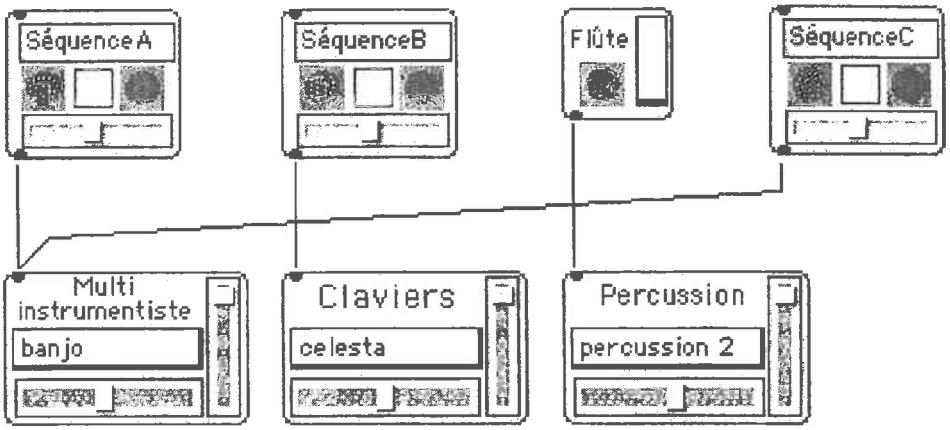


Figure 4. La composition de Tristan.

La composition de Tristan ressemble beaucoup à celle de François à deux détails près. D'abord, au lieu d'utiliser les objets déclencheur pour faire démarrer les séquences, Tristan cliquait sur le bouton vert des objets séquence pour les démarrer. Deuxièmement, il se sert de la flûte MIDI pour obtenir un son de percussion avant de démarrer la 3^{e} et dernière séquence.

La composition de Marie-Elaine

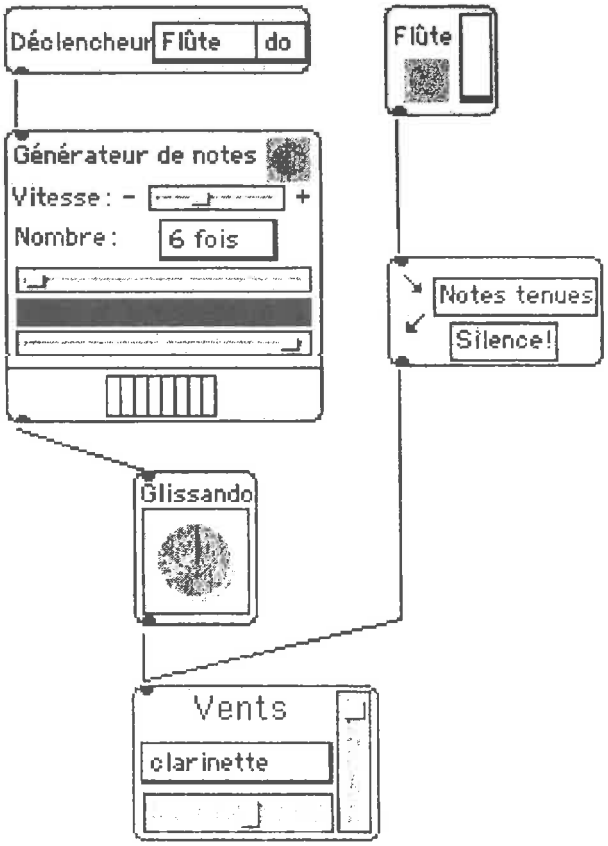


Figure 5. La composition de Marie-Elaine.

Marie-Elaine, contrairement à François et Tristan, n'emploie ni l'objet séquence ni le clavier MIDI. Elle utilise la flûte MIDI, à laquelle elle donne un son de clarinette, pour créer de longues notes tenues qu'elle module ensuite avec l'objet glissando. En fait, sa création est une longue improvisation dans laquelle sont manipulés en temps réel plusieurs objets comme « glissando » et « notes tenues ». Finalement — élément original de sa composition — elle utilise l'objet générateur de notes pour faire jouer six notes au hasard chaque fois que la flûte émet un *do*.

Trois types d'approches

Il est possible de dégager trois types d'approches quant à l'utilisation de CliMAX par les jeunes : l'approche instrumentale, l'approche par séquence et l'approche exploratoire[7]. L'approche instrumentale, utilisée par François et Tristan, se caractérise par une utilisation intensive d'un instrument (souvent le synthétiseur MIDI) qui se fait parfois au détriment de l'interface de CliMAX[8]. Le jeune se concentre sur l'instrument et passe de longues minutes à inventer une mélodie ou à essayer de reproduire une mélodie qu'il connaît déjà. Le jeune qui utilise cette approche utilise habituellement peu d'objets à l'écran.

L'approche par séquence, également utilisée par François et Tristan, se caractérise par une utilisation d'un ou de plusieurs objets séquence comme centre du processus de création. Le jeune enregistre une mélodie dans l'objet séquence pour ensuite faire rejouer la séquence, la modifier grâce à de nouveaux timbres, la transposer, etc. Après avoir enregistré sa mélodie dans l'objet séquence, le jeune abandonne généralement les instruments MIDI.

La troisième approche, qui est celle de Marie-Elaine, est l'approche exploratoire. Dans cette approche, le jeune explore l'environnement très librement, un peu comme s'il n'avait pas de but précis, et utilise pratiquement tous les objets de CliMAX l'un après l'autre. Souvent, les connexions entre les objets semblent faites de manière plus ou moins consciente. Finalement, le jeune ne semble pas avoir de préférence pour aucun des objets et il utilise incidemment les instruments MIDI.

CliMAX versus un logiciel séquenceur

Pour faire ressortir l'originalité de CliMAX, nous devons nous demander quel est l'élément essentiel qui le différencie des nombreux séquenceurs que l'on retrouve sur le marché. Remarquons tout d'abord que CliMAX inclut dans sa banque d'objets de traitement un séquenceur — l'objet séquence — possédant des fonctions élémentaires : enregistrement, lecture avec ou sans boucle, modification du tempo. Ultérieurement, il sera possible de complexifier l'objet séquence en y ajoutant d'autres fonctions comme une visualisation de la ligne mélodique (notation graphique de type « piano roll ») accompagnée des fonctions d'édition copier-coller[9].

---

[7] Il est à noter qu'une des recommandations de l'étude initiale (Daignault et Murray, 2001) est de valider ces trois approches avec un plus grand nombre de sujets, ce qui n'a pas été fait jusqu'à maintenant. Nous avons préféré nous concentrer sur la diffusion de l'environnement.

[8] Même si l'objet clavier (qui reçoit des notes du clavier MIDI) n'apparaît pas dans les compositions de François et Tristan, ces derniers avaient longuement manipulé cet objet pour enregistrer les mélodies utilisées. L'objet clavier a été supprimé à la fois pour alléger la présentation graphique des images et démontrer qu'il n'est plus nécessaire lorsque la musique est enregistrée dans un objet séquence.

[9] C'est l'objet pur de MAX appelé Detonate, figurant au cœur de l'objet de traitement séquence, qui permettra ces améliorations.

CliMAX se veut d'abord et avant tout une initiation à la programmation et à la création musicale. Le concept entrée-traitement-sortie, intimement lié au logiciel MAX, ne se retrouve pas, visuellement, de façon aussi explicite dans un séquenceur. Or, ce concept nous est très important en raison de sa valeur pédagogique. En effet, CliMAX permet à l'enfant de visualiser la création musicale au fur et à mesure qu'il construit celle-ci en assemblant les différents objets d'entrée, de traitement et de sortie. L'enfant peut ensuite, lors de la création, manipuler les objets de traitement et ainsi modifier le résultat sonore en temps réel[10]. C'est une des forces primordiales des systèmes temps réel comme MAX que l'on ne retrouve pas dans un séquenceur.

L'avenir de CliMAX

Pour l'instant, CliMAX ne dispose pas d'une grande quantité d'objets de traitement. Cependant, puisque l'environnement CliMAX repose sur le logiciel MAX, il est facile d'ajouter des objets à cet environnement. Nous entrevoyons dans un avenir rapproché la construction de trois types d'objets pouvant ajouter plusieurs possibilités à CliMAX. Premièrement, en plus du raffinement des objets déjà existants, nous pourrons construire d'autres objets de traitement permettant la modification des données MIDI (par exemple, un objet pour effectuer l'harmonisation d'une ligne mélodique, un objet arpégiateur, un objet gérant les messages de *control change*, etc.) Deuxièmement, nous pourrons introduire dans CliMAX le traitement audio en temps réel grâce aux objets MSP[11], objets qui permettront la modification de plusieurs paramètres sonores. Une attention toute particulière sera accordée lors de l'intégration de l'audio dans le même environnement que les sons MIDI. Il faut en effet préserver la simplicité de CliMAX étant donné qu'il s'adresse avant tout à des enfants. Finalement, la dimension multimédia de MAX pourra être exploitée en ajoutant des objets destinés au traitement des images et de la vidéo (par exemple, un objet permettant d'afficher un film au format QuickTime, un autre permettant l'affichage de couleurs qui se modifient en temps réel pendant que l'on joue, etc.) Toutes ces améliorations possibles de CliMAX devront faire l'objet d'études ultérieures.

En conclusion, nous croyons que CliMAX est un environnement viable de création musicale pour les jeunes et ce, pour plusieurs raisons. D'abord, à notre connaissance, c'est la première fois que MAX est utilisé en éducation musicale auprès des jeunes. Ensuite, les jeunes qui l'ont essayé se sont montrés enthousiastes. Ils ont apprécié l'interface visuelle de travail et le concept entrée-traitement-sortie de CliMAX. Enfin, parce qu'il repose sur MAX, langage de programmation graphique qui permet d'ajouter ou de supprimer des objets facilement, CliMAX présente l'avantage appréciable de pouvoir être modifié selon les besoins de chacun[12].

---

[10] Voir dans cet article la section intitulée « La composition de Marie-Elaine ».

[11] MSP est constitué de plusieurs objets qui s'ajoutent à MAX et qui permettent d'effectuer du traitement audio en temps réel. MSP est distribué par la même compagnie qui développe MAX : Cycling74 (www.cycling74.com).

[12] On peut se procurer gratuitement l'environnement de création musicale CliMAX dans la section « Technologie musicale » du site www.fmurray.com Il faut cependant posséder le logiciel MAX de la compagnie Cycling74 (www.cycling74.com). L'auteur est ouvert à tout commentaire susceptible d'améliorer CliMAX.

Bibliographie

Daignault, L., et F. Murray (2001). « Programmation et création musicale au secondaire ». *Recherche en éducation musicale* 19 : 3-22.

Déchelle, F. (1999). « jMax : un environnement de programmation pour l'interactivité et le temps réel ». *Interfaces homme-machine et création musicale*, sous la direction de H. Vinet et F. Delalande, 85-94. Paris : HERMES Science.

Higgins, W. (1992). « Technology ». *Handbook of Research on Music Teaching and Learning*, édité par R. Colwell, 480-497. New York : Schirmer Books.

Holland, S. (1989). « Artificial Intelligence, Education and Music : The Use of Artificial Intelligence to Encourage and Facilitate Music Composition by Novices ». Thèse de doctorat, The Open University, Milton Keynes.

MAX (version 3.5). Logiciel de programmation musicale. Cycling74 (www.cycling74.com), IRCAM (www.ircam.fr).

Appendice A
Description des objets de CliMAX

Catégories	Objets	Détails
Objets d'entrée	Clavier	Reçoit des informations musicales du clavier MIDI (synthétiseur).
	Flûte	Reçoit des informations musicales de la flûte MIDI. Un curseur indique la force du souffle.
	Déclencheur	Envoie un signal lorsqu'une note spécifique est jouée au clavier ou à la flûte MIDI (peut ainsi déclencher les objets séquence et générateur de notes).
Objets de traitement	Délai	Retarde la transmission des informations musicales selon une valeur déterminée par l'usager.
	Générateur de notes	Génère des notes au hasard selon certains paramètres (vitesse, nombre, étendue, échelle musicale).
	Glissando	Permet de modifier la variation de hauteur (*pitch bend*) des objets de sortie.
	Miroir	Transforme tout intervalle ascendant en intervalle descendant et vice-versa selon une échelle diatonique ou chromatique.
	Notes tenues	Permet la production de notes pédales.
	Transposition	Transpose les notes de –24 demi-tons à + 24 demi-tons en transposition tonale ou réelle.
	Séquence A, B et C	Permet d'enregistrer des données musicales, de les faire rejouer (avec ou sans boucle) et d'en modifier le tempo.
Objets de sortie	Claviers* Cordes* Vents* Synthétiseurs* Percussion* Multi-instrumentiste*	Permet de produire les sons de la famille des claviers. Permet de produire les sons de la famille des cordes. Permet de produire les sons de la famille des vents. Permet de produire les sons de la famille des synthétiseurs. Permet de produire les sons de percussion. Permet de produire les sons de toutes les familles d'instruments.
	Tutti	Envoie virtuellement les informations à tous les autres objets de sortie. Permet de modifier le volume global des autres objets de sortie.

* Ces objets de sortie possèdent un potentiomètre qui permet de régler leur volume et leur panoramique (position du son dans l'espace).

In Vitro

Implémentation d'agents autonomes en MAX/MSP

Mikhail Malt
mmalt@ircam.fr

Résumé : Nous exposerons dans cet article l'implémentation d'un modèle d'agents autonomes dans l'environnement MAX/MSP[1]. Cette implémentation faite dans le cadre du développement d'une installation temps réel, le projet « Traces », a par la suite donné lieu à une réflexion sur l'utilisation d'une certaine catégorie de modèles en composition musicale.

1. Introduction

Un des problèmes qui se posent lors de la conception d'installations temps réel, est le choix du, ou des modèles génératifs. La majeure partie des modèles utilisés de nos jours reposent sur l'utilisation de l'aléatoire comme simulation du choix.
Dans notre cas précis, nous voulions trouver un modèle qui puisse être plus qu'une simple simulation des choix possibles d'un compositeur, qui puisse être un modèle de contrôle du matériau musical, en nous rapprochant le plus possible d'un modèle génératif simulant une écriture musicale.
L'utilisation d'un modèle basée sur des agents autonomes nous semblait être une solution intéressante, puisqu'il propose une solution fondée sur l'évolution de petites entités musicales, qui interagissent entre elles de manière à dégager une structure complexe.
L'unique modèle d'agent disponible pour une application musicale était « boids » de Craig Reynolds, « implémenté » dans MAX par E. Singer. Cependant, comme le modèle avait une orientation graphique explicite, il ne permettait pas son extension à d'autres utilisations, ce qui nous a amené à développer la présente application.

2. Les agents autonomes

Depuis l'apparition de ce concept, au milieu des années soixante-dix dans le champ de l'intelligence artificielle distribuée, la dénomination d'« agent » est utilisée pour désigner soit des entités informatiques logiciels soit des entités informatiques matérielles (comme les robots). De ce fait il n'existe pas, de nos jours, un consensus général sur la définition « d'agent ».

Pour les besoins de notre travail, nous avons utilisé la définition donnée par M. Wooldridge:

« Un agent est un système informatique, situé dans un environnement et qui est capable d'actions autonomes dans cet environnement de manière à atteindre ses objectifs de conception » [Weiss 2000, 29].

[1] © Cycling74&Ircam, http://www.cycling74.com.

Nous ajouterons que « autonomie » signifie que l'agent doit être capable d'agir, réagir et interagir dans son environnement sans l'intervention d'autres systèmes (humains ou informatiques). Ceci nous amène au fait qu'un agent est une entité possédant : un état interne (dynamique) et des règles de comportement.

3. Pourquoi des « agents » ?

L'intérêt que peut avoir l'utilisation d'un modèle d'agents dans le contrôle d'évènements musicaux vient du fait qu'il fait partie des « modèles orientés individus » [Reynolds 2001]. Ces modèles sont des simulations basées sur les conséquences globales d'interactions locales entre les membres d'une population. L'action de contrôle se fait au niveau de l'individu, c'est-à-dire des règles qui commandent son comportement. Ceci revêt une importance majeure si on confronte ce type de modèle aux modèles statistiques. Bien que ces deux types de modèles aient comme vocation la génération d'informations de contrôle pour des situations globales, dans les modèles statistiques, le contrôle se fait sur des caractéristiques moyennes de la population, tandis que pour les « modèles orientés individus », le contrôle se fait plutôt au niveau des règles de comportement de l'individu.
Parmi les « modèles orientés individus », il existe aussi le modèle d'automate cellulaire, qui a été utilisé par plusieurs compositeurs [Miranda 1994]. Cependant, ce modèle n'étant pas capable de représenter des données complexes (ou multidimensionnelles), ni de représenter une « diversité » d'individus, puisque toutes les « cellules » sont semblables, n'était pas adapté à nos besoins. En revanche, le modèle d' « agent » permet une gestion d'espaces « n » dimensionnels et la représentation d'une grande diversité d'individus. De ce point de vue, ce modèle laissait entrevoir la possibilité de simuler d'une manière satisfaisante l'évolution d' « entités » musicales avec l'ensemble des paramètres nécessaires à leur représentation.

4. L'implémentation

4.1 Le cadre artistique

Cette implémentation fait partie d'un projet de concert/installation/performance appelé « Traces », avec la collaboration du contrebassiste français Jean Pierre Robert. Ce projet qui se veut aussi une performance/reflexion humaine/numérique sur le concept de vie, se divise en deux parties : la première, « In vivo » (la performance) et la deuxième, « In vitro » (l'installation). La phase « In vitro » propose, en temps réel et sans intervention humaine, l'évolution musicale du matériau généré par l'instrumentiste et la machine dans la première phase « in vivo », de la même manière qu'on cultive des microorganismes dans une « boîte de Petri[2] ».
Chaque être musical naîtra, vivra, interagira avec son environnement, sera influencé par lui, aura un nom (une généalogie se construira), se reproduira et mourra.

La surface musicale générée, devra avoir comme but la formalisation de certains aspects de notre écriture musicale. Cette écriture se fonde sur l'utilisation de petits gestes musicaux qu'évoluent dans le temps [Malt 1996].

[2] Petite boîte stérilisée qui permet de cultiver et d'observer facilement les bactéries.

4.2 Le cadre technique

L'implémentation de ce modèle s'est faite dans l'environnement MAX/MSP. L'environnement des agents est une fenêtre MAX (figure 1), et chaque agent est un « patcher» chargé dynamiquement (figure 2).

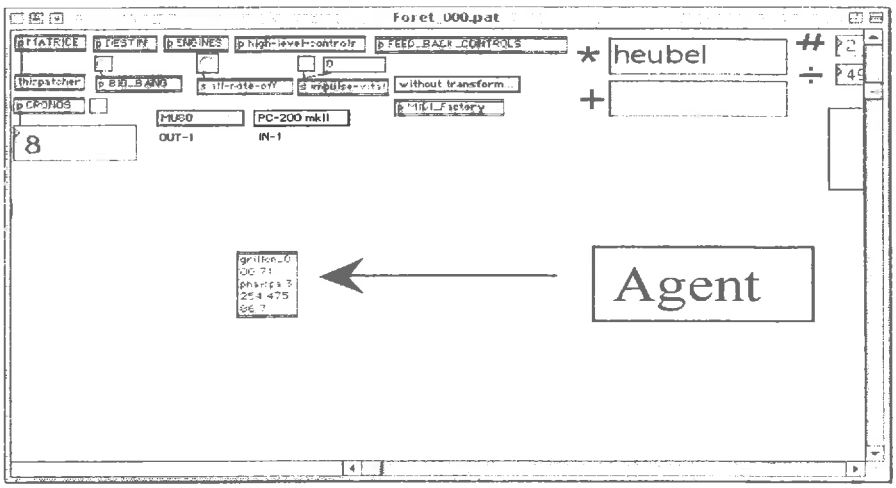


figure 1 : L'environnement

A sa naissance, chaque « individu » possède :

- Un nom, pour permettre entre autres son identification et de nommer les diverses variables pour l'envoi et la réception de messages « personnels ».
- Une position spatiale fixe, pour permettre de fixer des repères virtuels.
- Une note pivot, hauteur musicale pour déterminer le centre à partir duquel certains gestes seront joués.
- Un rayon d'écoute, pour restreindre l'appareil « cognitif » de chaque agent.
- Un âge maximal de vie, génétiquement déterminé ; au fur et à mesure qu'un agent vieilli, la probabilité de mort augmente.
- Un rôle, (un geste musical). Actuellement, il existe 10 rôles[3], chaque rôle étant associé à un geste musical (l'événement simple, la répétition, une « proto-mélodie », éléments de synthèse, échantillons, etc.).
- Un mode de comportement. Chaque agent est capable de connaître les agents qui sont à l'intérieur de son rayon d' « écoute », et de modifier certaines caractéristiques de son comportement et de son rôle en fonction de ses voisins. Par exemple, dans un mode de comportement « sociable », chaque agent tend à faire évoluer son geste musical vers la moyenne des « notes pivot » de ses voisins. Par contre, un mode de comportement « anti-social » fera évoluer l'agent à l'opposé de ses congénères.
- La possibilité de se reproduire. Le deuxième tiers de la vie maximale prévue, est la « période fertile », à ce moment chaque agent peux se reproduire en transmettant une partie de son nom, son l'âge maximale, son rôle et son comportement.

Chaque agent a un temps de vie et son action dans l'environnement est toujours transitoire. Chaque agent peut aussi se reproduire de manière à transférer une partie de son patrimoine génétique (son nom, son rôle, son matériau, son comportement et sa position

[3] Les rôles actuels sont en train d'évoluer de manière qu'il est possible que ce nombre change prochainement.

spatiale) à sa progéniture. Le résultat final de l'interaction entre les divers agents est la construction d'une « surface musicale ».

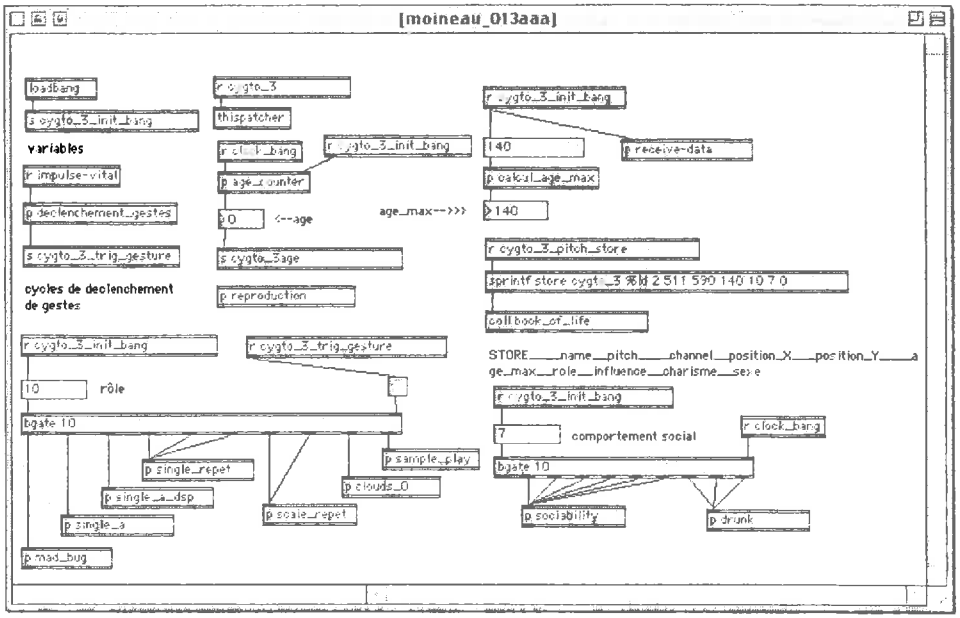


figure 2 : L'agent

Les « individus » sont générés de deux manières. La première est appelée « God_generated » et la deuxième est une reproduction asexuée, « Self_generated ». Le mode « God_generated » permet d'initialiser le système, de contrôler le nombre d'agents (évitant la disparition de la « société » créée, ou une surpopulation), et également d'introduire de nouveaux éléments (gènes) en favorisant la variété. Le mode « Self_generated ». permet la reproduction des agents de manière à perpétuer des noms, des rôles, des modes de comportement et des matériaux génétiques divers (MIDI et audio).

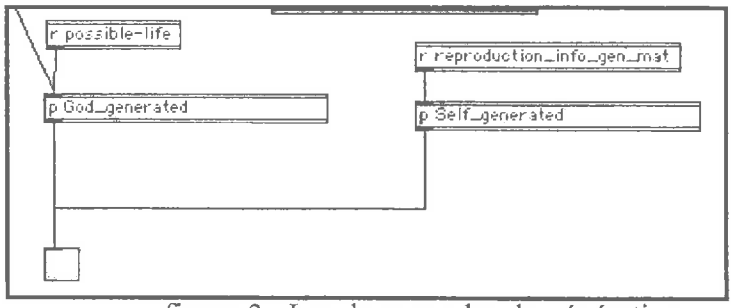


figure 3 : Les deux modes de génération

Ces deux modes de génération se fondent, d'un point de vue technique, sur la possibilité de l'environnement MAX de « charger » dynamiquement des « patchers[4] », à partir du message « newex » (voir figures 4, 5 et 6). Ce message prend comme paramètres la « commande » « newex », les positions relatives en pixels du patcher (X=511 et Y=590), des paramètres relatifs à la taille du patcher (largeur = 49 et taille de police, codée comme 196617 pour « geneva » 9), le nom du patcher (moineau_013) et la liste de paramètres que nous voulons

[4] Un « patcher » est une « encapsulation » d'un processus dans un objet graphique telle que p random_birth

associer à l'agent (note pivot, nom, canal MIDI, positions X et Y, âge maximal, index de rôle et index de comportement). Le « patcher » généré peut être vu figure 6.

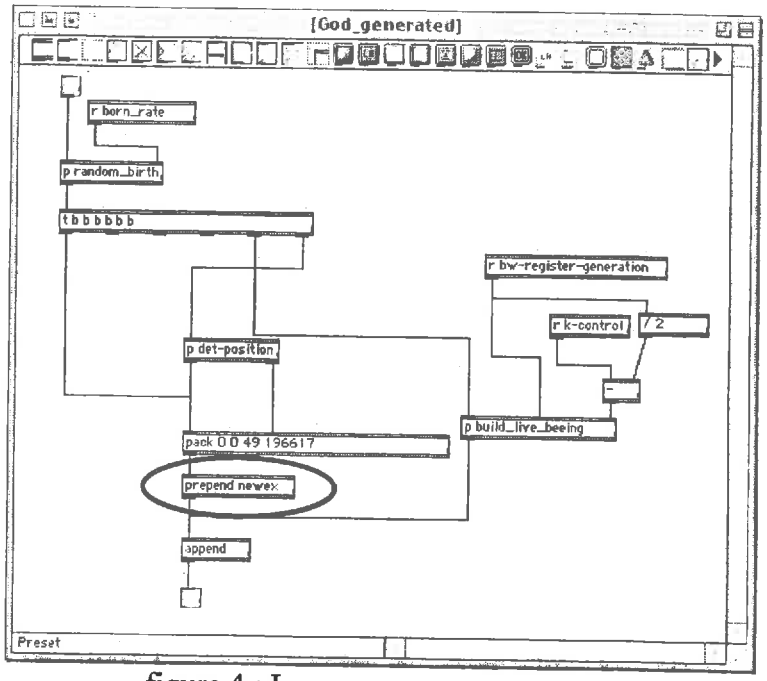


figure 4 : Le message « newex »

```
newex 511 590 49 196617 moineau_013 75 cygto_3 2 511 590 140 10 7 0 0
```

figure 5 : Le format de données pour « newex »

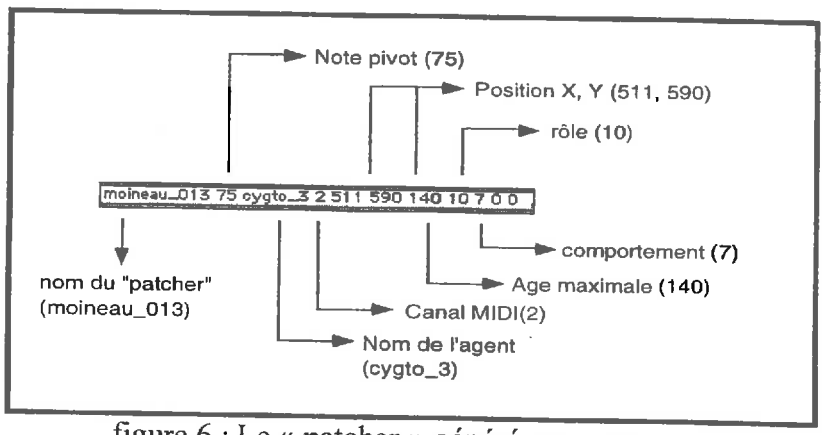


figure 6 : Le « patcher » généré par « newex »

5. Conclusions

Une des caractéristiques des modèles d'agents est leur pragmatisme implicite, c'est-à-dire, qu'ils sont très complexes[5] qu'il n'est possible d'apprivoiser que par la simulation numérique. Dans un travail où le matériau de base est un ensemble de processus, contrôlés par des modèles mathématiques et d'objets ayant presque une vie propre, l'ordinateur se montre être l'outil approprié pour gérer le grand nombre de variables et de relations.

D'après les expériences que nous avons menées, nous pouvons émettre l'hypothèse qu'une « surface musicale », pourra être vue comme un système auquel correspondra une dynamique instable, mue par une multiplicité de forces en interaction. La composition sera vue comme un processus en permanent mouvement, une recherche permanente de « sens » entre les divers niveaux de l'espace musical considéré, avec des moments de stabilisation, des moments de déstabilisation et principalement des phénomènes d'émergence. La notion d'émergence, qui est fondamentale dans ces expériences, exprime l'apparition d'un sens nouveau lors de l'agrégation d'éléments au sein d'un contexte. Ce sens nouveau est explicitement absent des éléments individuels. Il est une nouvelle caractéristique qui résulte de l'interaction entre ces éléments. C'est finalement penser la composition comme étant le produit d'un tissu de relations avec elle même et avec le monde qui l'environne. Relations aussi vivantes qui évoluent et se transforment pendant l'évolution de l'œuvre.

Le développent de ce travail se poursuit actuellement avec la recherche de représentations possibles pour les règles de comportement, l'implémentation de la reproduction sexuée, la représentation et le codage de divers matériau musicaux (MIDI et audio) pour étudier les possibilités de transmission génétiques et la recherche des liens entre les structures de contrôles générées et le sens musical produit.

[5] La Complexité de laquelle nous parlons n'a aucun lien avec le "compliqué", mais plutôt avec son sens étymologique: ***complexus***, ce qui est tissé ensemble. Des constituants divers associés, tissés ensemble pour produire une nouvelle unité.

Références

[Langton 1996] Cristopher C. Langton, *Artificial Life, an overview*, Cristopher C. Langton editor, MIT Press, 1996.

[Malt 1996] M.MALT, « Lambda3.99 (Chaos et Composition Musicale) », in *Troisièmes Journées d'Informatique Musicale JIM 96*, Ile de Tatihou, Normandie, France, 1996.

[Miranda 1994] E. R. Miranda, « Music composition using cellular automata », *Languages of Design*, Vol. 2, pp. 105-117, USA.

[Miranda 2000] E. R. Miranda (Ed.), *Readings in Music and Artificial Intelligence*, Contemporary Music Series Vol. 20, Harwood Academic Publishers, Amsterdam, 2000.

[Reynolds 1987] Craig W. Reynolds, "Flocks, Herds, and Schools: A Distributed Behavioral Model" in *Computer Graphics* 21(4) (SIGGRAPH 87 Conference Proceedings) pages 25-34. http://www.red.com/cwr/boids.html

[Reynolds 1999] Craig Reynolds, "Steering Behaviors For Autonomous Characters" *in Conference Proceedings of the 1999 Game Developers Conference*, pages 763-782. http://www.red.com/cwr/steer/

[Reynolds 2001] Craig Reynolds, « Individual-Based Models, an annotated list of links », http://www.red3d.com/cwr/ibm.html.

[Weiss 2000] Gerhard Weiss, *Multiagent Systems, a modern approach to distributed artificial inteligence*. Edited by Gerhard Weiss, MIT Press, 2000.

CAO et contraintes

Ch. Truchet, C. Agon, G. Assayag
IRCAM
1, place Igor Stravinsky
75 004 Paris
{truchet, agonc, assayag } @ircam.fr

Philippe Codognet
LIP6
8, rue du Capitaine Scott
75 015 Paris
Philippe.Codognet@lip6.fr

Résumé : Nous présentons deux méthodes complémentaires pour la programmation par contraintes en CAO. La première méthode est complète. C'est un solver par backtracking et propagation, qui a été importé dans le logiciel OpenMusic. La seconde, une méthode incomplète, est un algorithme de recherche locale dont nous montrons qu'il est particulièrement bien adaptée à la CAO. Les tests déjà réalisés sont très satisfaisants.

1 Introduction

Les problèmes de satisfaction de contraintes (CSP) sont un formalisme qui permet de modéliser les problèmes fortement combinatoires. Le problème est représenté par des variables, des domaines de valeurs pour chaque variable et des contraintes, relations sur les variables restreignant les valeurs qu'elles peuvent prendre. Une fois cette modélisation effectuée, la programmation par contraintes sert à résoudre ces problèmes, ie à affecter à chaque variable une valeur (instanciation de la variable) de telle sorte toutes les contraintes soient satisfaites pour ces valeurs.

Parmi les solvers actuels, on peut distinguer les méthodes complètes et les méthodes incomplètes. Les premières reposent sur une exploration exhaustive des domaines (backtracking), de complexité exponentielle. Elles peuvent être accélérées de plusieurs manières : forward checking (à chaque instanciation, les domaines sont réduits en fonction des variables déjà instanciées et des contraintes), back jumping (backtrack directement sur la variable en conflit avec la variable qui vient d'être instanciée), etc. Les méthodes incomplètes, elles, ne gardent pas en mémoire l'espace déjà visité dans les domaines. Elle ne terminent pas lorsque le problème n'a pas de solution, mais ont prouvé leur efficacité en temps de calcul sur nombre de CSP classiques (voir Section 4).

Nous présentons deux méthodes de résolution de contraintes adaptées à la Composition Assistée par Ordinateur. L'une est complète et assez classique, l'autre, appelée recherche adaptative [Cod00] est incomplète. C'est un algorithme de recherche locale.

Dans le domaine de l'informatique musicale, la programmation par contraintes a une place assez naturelle. Le cas de l'harmonie classique, déjà plusieurs fois traité, en donne un bon exemple. Les traités d'harmonisation donnent un ensemble de règles à respecter : mouvements contraires entre deux voix, pas de quintes parallèles, etc. Ils sont en quelque sorte entièrement déclaratifs. En musique contemporaine, domaine où l'informatique est déjà très utilisée, la notion de règle musicale reste, et quelques compositeurs utilisent déjà la programmation par contraintes.

Le précurseur dans ce domaine est Kemal Ebcioglu qui propose un système de composition de chorals dans le style de Bach [Ebc97]. Le même problème est traité par Tsang [TsA91], de manière encore assez lente, puis par Philippe Ballesta dont le système est basé Ilog-Solver [Bal98]. De même, François Pachet et Pierre Roy ont traité le cas de l'harmonisation classique [PAR95]. Implémenté en Backtalk, leur solver comporte une application musicale pour l'harmonisation automatique, en utilisant les structures musicales propres à la musique tonale.

La plupart de ces systèmes ont été validés et se comparent dans le cadre d'un problème très précis, l'harmonisation automatique, le plus souvent réduit au style des chorals de Bach. Lorsqu'ils proposent un framework plus général, il est lié à la musique tonale, très contrainte en matière de structure harmonique. Enfin, l'IRCAM propose déjà dans OM deux solvers de contraintes pour la musique contemporaine, Situation et PWConstraints.

Situation a été conçu par Camilo Rueda en collaboration avec le compositeur Antoine Bonnet [RUV97]. C'est un moteur de contraintes par forward checking, muni d'une interface graphique en OM. Situation est efficace mais présente plusieurs inconvénients. Le contrôle visuel du solver est limité. L'écriture des contraintes reste un exercice difficile de par la syntaxe ésotérique du langage. Situation ne permet pas d'approcher une solution (dans le cas d'un problème surcontraint par exemple). Enfin, il gère mal les contraintes globales. Il a essentiellement été validé dans le domaine harmonique.

PWConstraints (PWCS) a été conçu par Mikaël Laurson [Lau96]. Le moteur de résolution utilise les algorithmes classiques de forward checking et back-jumping. L'une des particularités de PWCS est de distinguer les règles standard des règles "heuristiques". Ces dernières sont ré-écrites non pas comme des prédicats, mais comme des fonctions renvoyant une valeur numérique. Elles expriment des préférences. Les solutions dont la valeur selon ces règles est la meilleure sont favorisées par le moteur.

Ces règles heuristiques approchent l'idée de hiérarchie sur les contraintes, ou de préférences, mais leur utilisation dans une stratégie de backtracking n'est pas toujours très heureuse. Par ailleurs, le concept de séquence utilisé par PW Constraints permet de décrire beaucoup de problèmes musicaux, mais pas tous. Enfin, PWCS traite mal les contraintes globales et l'aspect visuel est inexistant.

Notre but est de construire un système de programmation par contraintes à l'usage des compositeurs contemporains, dans le logiciel OM. Cela suppose d'abord de trouver un système général de spécification de contraintes musicales, adapté à l'interface visuelle intuitive d'OM. Comme le système est destiné à la composition contemporaine, il ne doit pas être marqué stylistiquement : bien sûr, on ne se limitera pas à la musique tonale, mais on ne doit pas non plus favoriser certaines catégories musicales (mélodie, harmonie, rythme, contrepoint, etc).

2 Cahier des charges

2.1 Expressivité

Nous avons commencé par consulter plusieurs compositeurs liés à l'IRCAM, qui avaient parfois déjà utilisé OM pour résoudre un problème de contraintes, voire déjà résolu un problème de contraintes empiriquement à la main ! Il en est sorti deux constats : les problèmes posés ne sont pas triviaux, et ils sont assez variés, à la fois dans la structure des objets utilisés et dans les contraintes. Les domaines sont en général entiers, mais pas toujours (fonctions, motifs rythmiques par exemple). Ils sont toujours finis. Les con-

traintes sont parfois simplement arithmétiques, plus souvent elles comportent des alldiff, des $\exists$, $\forall$, $\in$.

L'une des difficultés est donc de fournir une bibliothèque de contraintes raisonnablement expressive. On a en effet le choix entre deux extrêmes : écrire une librairie adaptée à un problème bien précis (contraintes harmoniques par exemple), facile d'utilisation. Mais alors elle risque de ne servir qu'à un compositeur. On peut aussi donner un solver qui accepte n'importe quelle contrainte, mais d'une part l'utilisateur devra faire une effort important pour écrire ses contraintes, d'autre part on risque de perdre en efficacité : on ne peut pas demander au compositeur un trop grand travail d'optimisation dans l'expression des contraintes (n'oublions pas qu'un compositeur n'a aucune raison de faire la différence entre poser une contrainte "strictement croissante" sur une suite, et poser une contrainte "alldiff", puis une contrainte "croissante", par exemple).

Pour éviter le premier écueil, nous avons commencé par la deuxième approche, de manière à avoir un langage de contraintes suffisamment expressif. Ainsi, nous pourrons ensuite proposer une bibliothèque de primitives plus élaborées.

2.2 Progressivité

En faisant appel à un solver de contraintes, le compositeur a bien sûr des attentes assez proches de celles de n'importe quel utilisateur (il veut une solution), mais il l'utilisera probablement un peu différemment. Nous avons constaté que le principe "attendre longtemps pour obtenir une solution exacte" n'était probablement pas le mieux adapté. En effet, il est probable qu'un compositeur, à qui on fournit une solution exacte, la retravaillera en fonctions de critères esthétiques, non formalisables. Par ailleurs, les solutions atypiques ou surprenantes ne sont pas à négliger, même si elles ne sont qu'approchées. Quant à l'attente, comme dans tout problème informatique, elle n'est pas souhaitable. Si on ne peut pas la réduire, on peut éventuellement la meubler, par exemple en affichant des solutions partielles, qui, si elles sont assez pertinentes, seront d'un réel intérêt pour l'utilisateur.

Nous avons décidé de distinguer les problèmes de génération d'un matériau musical à l'aide de contraintes, cas où une méthode complète est inévitable, des problèmes de résolution. Les problèmes de résolution seront traités avec une méthode incomplète, un algorithme de recherche locale par améliorations successives du résultat, plus adéquat aux besoins exposés ci-dessus.

3 Méthode complète

L'un des premiers travaux a été d'importer dans OpenMusic, développé par Gérard Assayag et Carlos Agon ([Ago98] et [ARL99]) un solver écrit en Common Lisp, Screamer ([SMA93]). Screamer ajoute à Common Lisp une forme d'indéterminisme via deux constructions, *either* et *fail*, qui introduisent le point de choix dans le langage. $(either\ e_1 ... e_n)$ évalue d'abord e_1, puis si l'évaluation de e_1 donne un *fail*, e_2, etc. Screamer propose en outre un solver avec propagation.

La version 4.0 d'OM possède maintenant une librairie Screamer, qui introduit toutes les fonctions permettant le backtracking et l'appel au solver. Pour l'instant, seules les fonctions faisant appel au backtracking ont été redéfinies pour s'intégrer à la programmation visuelle d'OM, la totalité du solver de Screamer sera ajoutée dans la prochaine version. L'indéterminisme posant problème pour un langage fonctionnel, nous avons ajouté au

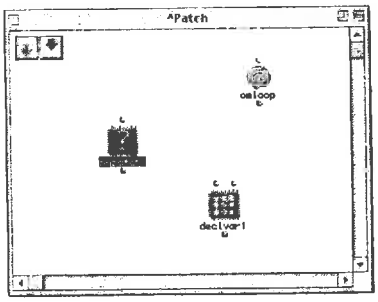


Figure 1: Exemples de patchs non-déterministes

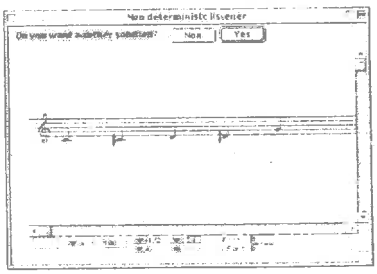


Figure 2: L'énumération des solutions a lieu dans le Nondeterministic Listener.

langage une nouvelle classe de patchs, qui correspondent aux fonctions déclarées indéterministes par Screamer. Ils sont notés avec un point d'interrogation, voir figure 1.

Lors d'une évaluation à la sortie d'un patch indéterministe, OM ajoute automatiquement l'une des trois fonctions de Screamer qui appellent une résolution. L'utilisateur peut, dans les Préférences, choisir le mode de résolution qui lui convient : une seule solution, toutes les solutions, énumérer les solutions. Dans les deux premiers cas, le résultat est directement renvoyé. Dans le deuxième cas, apparaît un Nondeterminitic Listener, voir figure 2, fenêtre affichant la solution courante, avec les mêmes éditeurs qu'OM (notamment musicaux, ce qui signifie que l'on peut écouter). Le Nondeterministic Listener permet aussi de choisir entre passer à une autre solution et garder la solution affichée.

Pour la déclaration de variables, l'utilisateur peut utiliser directement les primitives Screamer a-member-of et an-integer-between qui font ce que leurs noms indiquent. Nous avons ajouté list-of-members-of et list-of-integers-between, qui font également ce que leurs noms indiquent, et list-of-chord-in, plus intéressante car elle comporte une entrée optionnelle : celle-ci permet de poser des contraintes sur chaque accord de la séquence d'accords, ceci pour gagner du temps de calcul. Ces cinq fonctions sont identifiées par une icône ad-hoc, voir figure 3. Dès qu'une de ces quatres fonctions est appelée dans un patch, le patch est classé non-déterministe. Les contraintes sont écrites comme des prédicats OM, et sont traduites en Screamer à l'évaluation. Elles sont passées au solver via une fonction apply-cont, voir figure 3. Elles s'intègrent donc de manière naturelle dans le langage visuel.

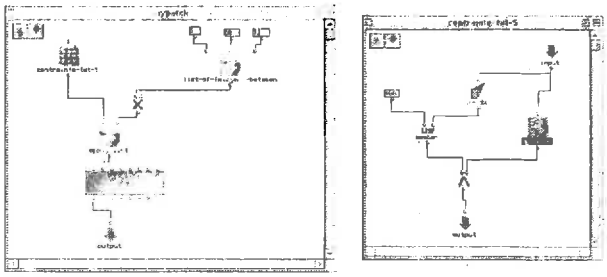


Figure 3: Un exemple de patch utilisant Screamer pour calculer des valeurs midi pour un chord-seq. A droite, la contrainte utilisée, exprimée comme un préicat OM. Elle impose que toutes les notes soient différentes, et que l'accord contienne une tierce majeure.

4 Méthode incomplète

4.1 Description

L'algorithme de recherche adaptative est proposé par Philippe Codognet (LIP6) [Cod00], qui l'a testé sur des problèmes classiques (N-reines et carrés magiques). Le problème est donné sous la forme d'un CSP, avec variables, domaines finis associés, et contraintes sur les variables. La recherche adaptative fait partie des algorithmes de recherche locale, tel que GSAT [SLM92], qui tirent profit de la représentation du problème en CSP, même s'ils se distinguent des techniques classiques de résolution. De tels algorithmes ont largement prouvé leur efficacité sur des problèmes comme celui du voyageur de commerce, des N-reines, etc.

La représentation d'un problème est celle du format CSP, avec $V_1...V_n$ les variables, $Dom_1...Dom_n$, les domaines finis associés, et $C_1...C_p$ les contraintes portant sur les variables. Les contraintes sont écrites sous la forme d'une relation logique entre les variables. On notera V_i la variable et v_i une valeur de V_i.

Le principe en recherche locale est de guider la recherche de solution par une mesure de la qualité d'une configuration. On peut résumer grossièrement ce type d'algorithme par : intialisation aléatoire, puis itérativement exploration d'un voisinage, recherche d'une meilleure configuration, remplacement. Cela suppose d'avoir une mesure de la qualité de la configuration courante, ce qui est fait en représentant les contraintes par une fonction de coût, qui sert à la recherche d'une meilleure configuration.

L'algorithme de recherche adaptative fonctionne sur ce principe, mais en affinant la notion de coût. Il s'agit de tirer le maximum d'information à partir des contraintes, au niveau de chaque variable V_i et non plus de la configuration $V_1...V_n$. Cela permet de sélectionner à chaque pas la variable la plus mauvaise. Nous remplaçons l'étape "exploration du voisinage" par le calcul des coûts de chaque variable, la sélection de la plus chère, et l'exploration du domaine de cette variable pour trouver une meilleure valeur.

Les différentes fonctions utilisées sont représentées ci-dessous :

$$f_{v_i,C_j}(v_i\,,\,(v_1...v_n)) \overset{somme\ en\ C_j}{\longrightarrow} f_{v_i}(v_i\,,\,(v_1...v_n)) \overset{somme\ en\ v_i}{\longrightarrow} f_{total}((v_1...v_n))$$

$f_{v_i,C_j}(v_i\,,\,(v_1...v_n))$ n'est pas directement utilisée dans l'algorithme, elle sert à la définition d'une grammaire de contraintes (voir ci-dessous). $f_{v_i}(v_i\,,\,(v_1...v_n))$ représente le poids d'une variable dans la configuration courante. Avec par exemple trois variables V_1,

V_2 et V_3, et deux contraintes $V_1 = V_2$ et $V_2 = V_3$, on peut choisir $|V_1 - V_2|$ pour V_1, $|V_1 - V_2| + |V_2 - V_3|$ et V_2 et $|V_2 - V_3|$ pour V_3. Cette fonction sert à sélectionner la variable la plus chère $V_{plus-chere}$ à chaque pas. La dernière fonction, $f_{total}((v_1...v_n))$ représente le poids de la configuration courante, dans notre exemple $2 * |V_1 - V_2| + 2 * |V_2 - V_3|$. Elle est utilisée pour déterminer une meilleure valeur sur le domaine de $V_{plus-chere}$.

Dans le cas où la variable $V_{plus-chere}$ n'a pas de meilleure valeur (aucune substitution de $v_{plus-chere}$ par une autre valeur du domaine ne permet de diminuer le coût global), l'algorithme boucle (minimum local de f_{total} sur l'axe $V_{plus-chere}$). Pour éviter de boucler, nous utilisons une mémoire adaptative, à la manière du Tabu Search [AaL97]. Si l'exploration du domaine $Dom_{plus-chere}$ ne donne pas de meilleure valeur, $V_{plus-chere}$ est marquée Tabu et ne pourra être modifiée pendant un nombre fixé d'itérations.

Il se peut aussi que l'on arrive dans un minimum local de f_{total}, sur tous les axes $V_1...V_n$. Dans ce cas, toutes les variables seront successivement marquées Tabu. Nous choisissons alors de ré-initialiser aléatoirement toutes les variables.

En réalité, il peut être plus efficace de ne pas tester toutes les variables lorsque l'on arrive dans un tel minimum, d'après [GSK98]. On peut par exemple fixer un nombre maximum de variables à tester, puis réinitialiser certaines ou toutes les variables. On a ainsi deux paramètres à fixer avant la résolution. Nous cherchons actuellement des valeurs optimales pour ces paramètres. Dans les résultats donnés en section 5, c'est la méthode décrite plus haut qui est utilisée (test de toutes les variables, réinitialisation de toutes les variables).

4.2 Algorithme

1. Initialisation aléatoire

   Repeat

2. Calcul des coûts de toutes les variables, sauf sauf celles marquées Tabu. Sélection de la plus chère, $V_{plus-chere}$.

3. Test du coût global en remplaçant $V_{plus-chere}$ par toutes les valeurs de son domaine. Sélection de v' la meilleure.

4. Si à la fin de l'exploration du domaine, aucune valeur n'améliore le coût global, alors $V_{plus-chere}$ est marquée tabu.

5. Si toutes les variables sont tabu alors réinitialisation aléatoire.

   Until l'erreur globale est inférieure à ϵ.

Si le problème n'a pas de solution, l'algorithme ne termine pas. On peut éviter cela en fixant un nombre maximum d'itérations. Nous avons implémenté cet algorithme en Lisp, avec des structures de données qui permettent d'utiliser les résultats directement dans OM (listes et listes de listes).

4.3 Avantages

Dans le cas des problèmes musicaux, cet algorithme présente plusieurs avantages. D'abord, il répond à notre objectif de progressivité. Nous avons ajouté dans l'implémentation une variable $affichage$, nulle par défaut. Cette variable fonctionne comme un seuil : dès que

l'erreur globale de la configuration est inférieure à $affichage$, le meilleur minimum local et son erreur sont affichés. Les solutions approchées sont traitées de la même manière. La condition d'arrêt est que la somme des coût pour toutes les variables soit inférieure à un nombre fixé, soit ϵ. De cette manière, on obtient les solutions exactes en prenant $\epsilon = 0$, et des solutions approchées pour $\epsilon > 0$.

La représentation des contraintes par des coûts apporte une souplesse supplémentaire au programme. En effet, on peut donner plus d'importance à certaines contraintes en pondérant leurs fonctions de coûts, et inversement laisser une tolérance sur d'autres. Ainsi, si l'on part d'un ensemble de contraintes $C_1...C_n$ de coûts $f_{C_1}...f_{C_n}$, et que l'on souhaite une solution exacte sauf pour C_j, on prendra comme fonction de coût totale $f_{total} = M * (\sum_{k \neq j} f_{C_k}) + f_{C_j}$, et l'on arrêtera le calcul dès que le coût est strictement inférieur à M (en fixant $\epsilon = M$).

Enfin, le cas de contraintes globales portant sur un ensemble de variables se traite facilement, ce qui représente un progrès par rapport à Situation et PWCS. Une contrainte globale a pour seule particularité d'avoir une fonction de coût constante sur les V_i. Cela ne gène en rien la résolution.

4.4 Grammaire des contraintes et génération des coûts

Il est évidemment hors de question de demander à l'utilisateur de trouver lui-même la fonction de coût associée à chaque contrainte. Nous avons donc écrit une fonction G de traduction, qui passe de la contrainte à sa fonction de coût (ici, $f_{v_i C_j}$, dont on déduit les autres). Les contraintes sont écrites en Lisp.

La grammaire des contraintes s' écrit :

$C ::= (t = t) \mid (t \leq t) \mid (t < t) \mid (t \in t) \mid (C \wedge C) \mid (C \cup C) \mid (alldifft)$
$\mid (\exists t \in t,\ C(t)) \mid (\forall t \in t,\ C(t))$

$t ::= n \mid (f\ t...t)$

avec n un entier et f une fonction de Lisp.

Dans un but d'optimisation non encore implémentée, on choisit de préférence une fonction raisonnablement continue (sur les réels) dans chaque classe, et même affine par morceaux, ce que l'on obtient facilement par construction de G (hors $alldiff$). La génération d'une fonction de coût se fait selon les règles suivantes. Les termes ne sont pas modifiés : $G(t) = t$. Pour une contrainte C :

$G(t_1 = t_2) = |t_1 - t_2|$
$G(t_1 \leq t_2) = max(0, t_1 - t_2)$
$G(t_1 < t_2) = max(0, 1 + t_1 - t_2)$
$G(t_1 \in t_2) = min_{t \in t_2}(|t_1 - t)|)$
$G(C_1 \wedge C_2) = max(G(C_1), G(C_2))$
$G(C_1 \vee C_2) = min(G(C_1), G(C_2))$
$G(alldiff(t_1...t_n)) = Card\{t_i = t_j, i < j \leq n\}$
$G(\forall t_1 \in t_2\, C(t_1)) = max_{t_1 \in t_2} G(C(t_1))$
$G(\exists t_1 \in t_2\, C(t_1) = min_{t_1 \in t_2} G(C(t_1))$

4.5 Problème du alldiff

Le $alldiff$ est difficile à représenter par une fonction de coût. La méthode grossière qui consiste à choisir pour $f_{alldiff}(V_i)$ la fonction caractéristique de $\{V_j = V_i\ j \neq i\}$ peut être un peu améliorée en prenant pour $f_{alldiff}(V_i)$ le nombre de j tels que $V_i = V_j$, mais ce n'est

pas très satisfaisant. Evidemment, n'importe quelle fonction de coût pour un $alldiff$ aura cette forme.

Dans le cas où les domaines des $V_1 ... V_n$ (variables sur lesquelles on pose le $alldiff$) sont égaux, et de cardinal t, une solution est de changer le domaine. Cette technique est utilisée par P. Codognet dans [Cod00], pour le problème des carrés magiques. Au lieu de modifier les V_i n'importe comment dans Dom_i, puis de calculer le $alldiff$, on travaille sur les transpositions qui échangent V_i avec une autre variable. De cette manière, si à l'initialisation le $alldiff$ est respecté, il le sera à chaque pas également et on l'obtient gratuitement en temps. Cela vaut par exemple pour les problèmes portant sur des séries (au sens musical : permutation des douze notes de la gamme). C'est ce principe que nous avons utilisé pour le problème des all-intervals series.

Nous envisageons d'étendre cette idée au cas où l'on a $alldiff$ sur m variables, sur le même domaine de taille t (avec évidemment $m \leq t$). L'idée est de retirer du domaine les valeurs instanciées. Cela peut se faire facilement sans modifier le solver, par exemple en ajoutant $t - m$ variables fantômes, dont les coûts seront toujours nuls, qui ne pourront donc être choisies pour une modification. Mais dans le cas où m est beaucoup plus petit que t, nous avons constaté qu'il était plus efficace de réduire directement les domaines. Lors de l'étape "recherche d'une meilleure valeur, on ne parcourt que la partie du domaine consistante pour le $alldiff$, ce que l'on peut voir comme un mini forward-checking.

5 Expériences

Nous avons effectué une série d'expériences dans OM pour comparer le temps de réponse du système de backtracking de Screamer et d'un prototype de solver par recherche adaptative. L'ordinateur est un Mac G4. Le temps de calcul donné ici ne comprend pas le garbage collecting. Pour la recherche adaptative, le nombre d'itérations désigne le nombre d'appels à la fonction principale. Nous donnons ici les résultats moyens pour dix résolutions. Signalons que l'écart-type est assez élevé. Pour le backtracking, le nombre d'itérations désigne le nombre de backtracks. Nous avons pris comme tests trois problèmes musicaux. Le premier est un problème musical classique. Le deuxième a été posé par Fabien Lévy, compositeur. Le troisième vient de Mauro Lanza, compositeur. Ces trois problèmes donnet un panorama de ce que l'on doit pouvoir traiter.

5.1 All-intervals series

Il s'agit de trouver une permutation σ des n premiers entiers, telle que les $|\sigma_{i+1} - \sigma_i|$ soient une permutation des $n - 1$ premiers entiers. C'est un problème musical classique, décrit notamment dans [MOS74]. Avec $n = 12$, on obtient une série au sens musical. La contrainte sur les différences successives impose que chaque intervalle soit aussi entendu exactement une fois lorsque l'on joue la série. Le problème a une solution triviale avec $1\,n\,2\,(n-1)\,3$ etc. On cherche bien sûr les autres solutions. Les all-intervals series ont été utilisé comme test pour Ant-P-Solver, de C. Solnon [Sol00], dont nous indiquons les résultats dans ce tableau. Pour la recherche adaptative, on utilise le $alldiff$ par transpositions comme défini ci-dessus. Ces résultats peuvent aussi être comparés à ceux d'Ilog Solver. Pour 20 entiers, Ilog Solver met plus d'une heure.

Nombre d'entiers	Backtracking	Adaptative	Ant-P-Solver
8	6 min	< 0.01 s	
10	> 1 h	0.02 s	0.0 s
12	> 1 h	0.1 s	0.1 s
14	> 1 h	0.2 s	0.5 s
16	> 1 h	2 s	2 s
18	> 1 h	5 s	3.7 s
20	> 1 h	18 s	10.4 s

5.2 Suite d'accords avec note commune

On considère une suite d'accords représentés par leurs valeurs midi. On souhaite avoir une ou plusieurs notes communes entre deux accords successifs. Les accords ont en plus une structure particulière : dans un même accord toutes les notes doivent être équidistantes en fréquence. On utilisera donc évidemment une représentation avec la fondamentale, l'intervalle entre deux notes, et le nombre de notes par accord.

Les tests ont été faits pour des suites de 20, 30 et 40 accords, avec à chaque fois des accords 4 notes et de 8 notes. Les contraintes sont d'une part que tous les accords soient différents, d'autre part qu'il y ait une et une seule note commune entre deux accords successifs. Les variables ont un domaine de taille 100.

		Backtracking		Adaptative	
Accords	Notes	Itérations	Temps	Itérations	Temps
20	8	218	4 s	46	1,8 s
20	4	355	1,8 s	42	1,3 s
30	8	460	17 s	63	1,9 s
30	4	569	6 s	53	2 s
40	8	780	52 s	56	1,7 s
40	4	948	17 s	53	2 s

5.3 Rythmes sans simultanéité

Il s'agit de composer un quasi-canon rythmique. On a n voix jouant simultanément. La i-ième voix répète un motif rythmique de période T_i, formé d'un ensemble d'onsets. L'unité de temps est donnée, de sorte que les onsets sont représentés par des entiers. Il s'agit de trouver les motifs pour que jamais deux voix n'aient deux onsets simultanés, et ce pendant une durée D fixée, $D \leq ppcm(T_1...T_n)$. Les T_i sont en général choisis premiers entre eux, pour avoir des séquences suffisamment longues. De ce fait, le nombre de variables est généralement élevé. La figure 4 montre une solution approchée, avec des T_i de 8, 14 et 12 unités de temps, ici la double croche. La figure 5 est une solution exacte.

En réalité, le compositeur souhaite éventuellement avoir non pas zéro onset simultané, mais le minimum possible, sachant qu'en fonction du nombre de voix et d'onsets le problème peut ne pas avoir de solution (de manière évidente, dès que les nombres d'onsets deviennent trop grands). Il accepte donc des solutions approchées.

Pour la résolution, les paramètres à choisir par l'utilisateur sont le nombre de voix, la durée D, et la densité d'onsets, qui représente le rapport entre le nombre d'onsets joués sur la longueur totale jouée. Par exemple, pour une densité de deux, on entendra en moyenne un onset toutes les deux unités de temps. Les tests ont été effectués avec une densité de 2. La durée choisie est 128 pulsations. Les longueurs des voix sont dans l'ordre

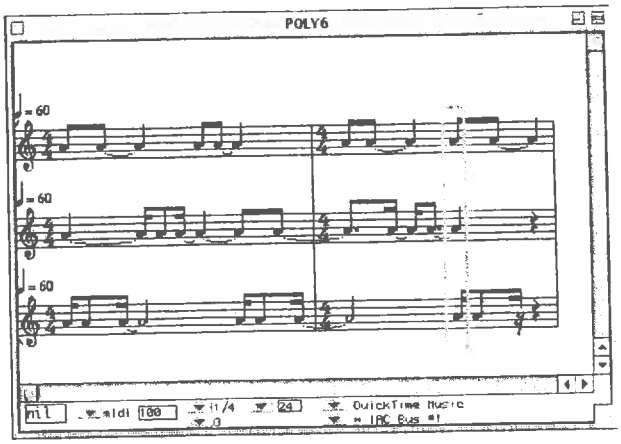


Figure 4: Une solution approchée. L'erreur est encadrée en gris, les voix inférieures et supérieures frappant un onset sur ce temps.

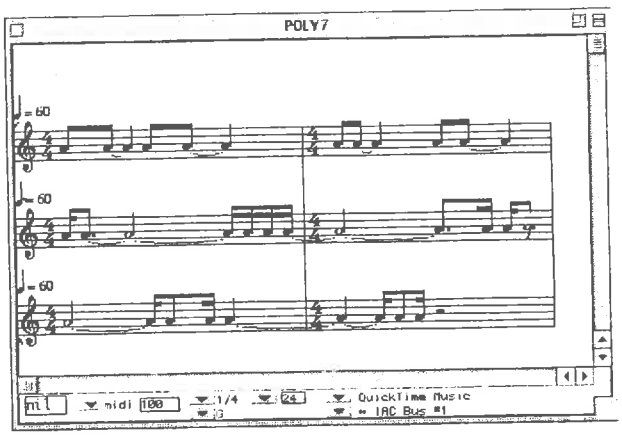


Figure 5: Une solution exacte du même problème.

19, 23, 29, 31, 37 et 43.

	Backtracking	Adaptative	
Nombre de voix	Temps	Itérations	Temps
3	> 1h	350	12 s
4	> 1h	463	22 s
5	> 1h	923	58 s
6	> 1h	1208	108 s

6 Conclusion

Les premiers résultats de recherche adaptative en musique sont encourageants. Bien sûr, le temps de calcul est raisonnable. Mais les particularités de la recherche adaptative (incomplétude, progressivité) en font un algorithme particulièrement bien adapté aux problèmes musicaux. Nous pensons élargir la gamme des problèmes tests.

Il reste cependant à améliorer l'algorithme et son implémentation. Nous envisageons notamment de tester le programme avec d'autres stratégies que la réinitialisation aléatoire en cas de minium local, d'implémenter un $alldiff$ plus efficace comme décrit ci-dessus,

d'améliorer la partie tabu de l'algorithme et enfin d'utiliser le caractère affine des fonctions de coût pour optimiser la recherche du meilleur candidat.

Viendra ensuite le moment d'écrire des primitives plus élaborées pour le langage de contraintes, et de réfléchir à une intégration visuelle satisfaisante de la recherche adaptative dans OM.

References

[AaL97] E. H. L. Aarts and J. K. Lenstra. Local search in combinatorial optimization. John Wiley and Sons, 1997.

[Ago98] Augusto Agon. An environment for computer assisted composition. Thèse de doctorat, IRCAM-UniversitÈ de Paris VI, 1998.

[ARL99] Gérard Assayag, Camilo Rueda ans Mikael Laurson, Carlos Agon, and Olivier Delerue. Computer assisted composition at ircam : Patchwork & openmusic. Computer Music Journal, 1999.

[Bal98] Philippe Ballesta. Contraintes et objets, clefs de voûte d'un outil d'aide à la composition. Editions Hermès, 1998.

[Cod00] Philippe Codognet. Adaptive search, preliminary results. 2000.

[Ebc97] Kemal Ebcioglu. An expert System for Harmonizing Chorales in the style of J.-C. Bach. AAAI Press, 1997.

[GSK98] Carla P. Gomes, Bart Selman, and Henry Kautz. Boosting combinatorial search through randomization. 2000.

[Lau96] MikaŚl Laurson. Patchwork : a visual programming language and some musical applications. 1996.

[MOS74] Robert Morris and Daniel Starr. The structure of the all-interval series. Journal of Music Theory, 13(2), 1974.

[PAR95] FranlJois Pachet and Pierre Roy. Integrating constraint satisfaction techniques with complex object structures. pages 11–22, Décembre 1995.

[RUV97] Camilo Rueda and Franck Valencia. Improving forward checking with delayed evaluation. 1997.

[SLM92] Bart Selman, Hector Levesque, and David Mitchell. A new method for solving hard satisfiability problems. pages 440–446, 1992.

[SMA93] Jeffrey Mark Siskind and David Allen McAllester. Nondeterministic lisp as a substrate for constraint logic programming. pages 133–138, 1993.

[Sol00] Christine Solnon. Ant-p-solver : un solver de contraintes à base de fourmis artificielles. pages 189–204, 2000.

[TsA91] C. P. Tsang and M. Aitken. Harmonizing music as a discipline of constraint logic programming. pages 61–64, 1991.

The Geometrical Groove: rhythmic canons between Theory, Implementation and Musical Experiment

Moreno ANDREATTA, Thomas NOLL, Carlos AGON and Gerard ASSAYAG

Keywords: Rhythmic canons, tessellation, modulation, augmentation
Domain: Formalisation and representation of musical structures

Introduction

During the second half of the twentieth century, algebraic methods have been increasingly recognised as powerful approaches to the formalisation of musical structures. This is evident in the American music-theoretical tradition as well as in the European formalised approach to music and musicology. We mention the mathematician and composer Milton Babbitt, the Greek composer Iannis Xenakis and the Roumanian theoretician and composer Anatol Vieru, that gave important impulses to the subject of our paper (Babbitt, 1960; Xenakis, 1971; Vieru, 1980). We also mention Gerald Balzano's original contribution (Balzano, 1980) and Dan Tudor Vuza's model of Vieru's modal theory (Vuza, 1982-), as well the approaches of Guerino Mazzola (Mazzola, 1990), Harald Fripertinger (Fripertinger, 1991) and Marc Chemillier (Chemillier, 1990), who opened the path to a generalisation and implementation of algebraic properties of musical structures.
This paper especially deals with the implementation of Vuza's model of periodic rhythm in OpenMusic, an open source visual language for composition and music analysis developed by IRCAM. This has been done in a specific library OMCanons as a part of a more general library called Zn, entirely based on the algebraic properties of finite cyclic groups and their applications to music. A complete catalogue of intervallic structures (up to transposition) is the starting point for a classification of intervallic structures by means of musically interesting algebraic properties (Olivier Messiaen's limited transposition property, Milton Babbitt's all-combinatoriality, Anatol Vieru's partitioning modal structures etc.), their generalisation for any n-tempered system and reinterpretation in the rhythmic domain.
In this article we deal with rhythmic canons of various kinds in order to show how the new OM-library can be used. However, there is a strong connection between well-known phenomena in n-tempered systems and n-cyclic time. Just to mention one example, Messiaen's limited transpositional modes are related to non-maximal-category canons (see below).

Rhythmic Canons Tiling the Space

The present essay focuses on the implementation of a family of rhythmic canons having the property of tiling musical time space. Before describing them in terms of an abstract model of cyclic time, we view them as they may appear within a musical composition, in the 'free' linear time, which has no cyclicity. Like in a melodic canon, one has several voices that may enter one after the other until all voices are present. As in the case of a melodic canon all voices are just copies of a ground voice that is suitably translated in the time axis. For simplicity - but yet with respect to linear time - we suppose here, that all voices are extended ad infinitum. We further suppose that the ground voice is a periodic rhythm that we will call the 'inner rhythm'. Following Vuza's definition, a periodic rhythm is an infinite subset R of the rationals Q (marking the attacks points, or onsets) with $R = R + d$ for a suitable period d. Furthermore, R is supposed to be locally finite (i.e. the intersection of R with every time segment [a, b] is finite). The period of a periodic

rhythm R is the smallest positive rational number dR satisfying R = R + dR. We also mention another important characteristic of a periodic rhythm - its pulsation pR. It is defined as the greatest common divisor of all distances between its attack points. Obviously, the pulsation pR of a periodic rhythm always divides its period dR.

In order to include the idea of tiling time space into the definition of a rhythmic canon, we need a further preparation: For each voice V of a canon we consider all rational numbers s such that V = R + s. The collection S of all these translations for all voices is itself a periodic rhythm (with period dS dividing dR) that works in fact as a 'meter' for the canon. Note that R and S may have different pulsations pR and pS. The pulsation of a canon is hence to be defined as the greatest common divisor p of the pulsations pR and pS.

The ratio n = dR/p is central in order to switch from linear time, modelled by rational numbers to circular time modelled by residue classes of integers. The transition goes as follows: Let r denote a fixed attack point within the inner rhythm R (If only one canon is being considered one can always suppose r = 0). Then each attack point in any of the voices has the form r + t p for a suitable integer t, i.e. the whole canon is contained in the sublattice r + p Z of Q. Because everything is periodic with period dR, we can work with classes of points in linear time and identify them with cyclic time points. Mathematically, one works with the factor space (r + p Z) / dR Z which may be identified with Z/nZ where n = dR/p.

From now on, we consider the whole canon within Z/nZ. We study the projections of R and S as well as those of the voices V (using the same notation) and formulate additional conditions in order to characterise rhythmic canons. For practical reasons we also allow cycles n that are multiples of dR/p.

Consider two subsets R and S of Z/nZ, the inner rhythm and the outer rhythm. Moreover consider the Voices Vs = R + s, where s runs through S. The pair (R, S) is said to generate a rhythmic tiling canon with the voices Vs if the following conditions are fulfilled:

1) The voices Vs cover entirely the cyclic group Z/nZ. With respect to the linear time this means that the canon is completely tiling musical time space at the (regular) pulsation p.

2) The voices Vs are pairwise disjoint. This means that the voices are complementary.
Periods dR and dS and pulsations pR and pS of R and S are also defined in Z/nZ. Among all canons having the properties 1) and 2) there is the special class of Regular Complementary Canons of Maximal Category, shortly RCMC-Canons (Vuza, 1995). They have the following additional property:

3) The periods pR and pS coincide.
Formally speaking, a RCMC-Canon is a factorisation of a cyclic group Z/nZ into two non periodic subsets (where a subset M of Z/nZ is said to be periodic if there exists an element t in Z/nZ such that t + M = M).

This transition from free linear time to cyclic time, that has been implemented together with all numerical invariants attached to a rhythm (period, number of attacks in a period, pulsation of a rhythm, ...), reduces the difficulty of many operations on rhythms that are connected with the construction of canons. One operation, called 'composition', is particularly relevant in this context. It represents Vuza's translation, in the rhythmic domain, of Anatol Vieru's composition of modal structures. We recall that for Vieru, a modal structure is a transposition class of any subset of the

cyclic group Z/nZ. Composing two modal structures simply means to take the union of the transpositions of the first one by the intervals determined by the second structure (or vice versa, because of the commutativity of transpositions). This operation, which is in fact a generalisation of Boulez' 'multiplication d'accords', leads to the formalisation of the canons construction process, as the composition of two rhythmic structures, respectively the inner class and the outer class (replacing Vuza's ground and metric classes). The first one gives the rhythmic pattern of a voice and the second one defines where the other voices have to enter. A special case consists of composing a rhythm with a regular rhythmic structure. This operation is called 'condensation'. In the case of canons, condensation is a powerful operation in interaction with another one: the exchange of the roles of inner and outer classes. This combination allows starting with any canon and ending up with a canon of maximal category. The implementation of this minmax-condensation algorithm (Vuza, 1995) is particularly useful for composers who are interested in the collection of all canons sharing, in some sense, the same genetic information of a particular prototype. For RCMC-canons, the implementation of Vuza's algorithm on OpenMusic enables to calculate, for any period n, all possible inner and outer structures associated with. It offers to understand the relationships between a period and the number of voices for such a canon. For example, the smallest RCMC-canon has a period equal to 72 and a number of voices equal to 6. This is the consequence of the algebraic property that no cyclic group smaller than Z/72Z can be 'factorised' in two non-periodic subsets. We now use this example to explain the idea of canon modulation.

Canon Modulation

In a compositional situation one might intend to work with more than just a single canon.
In that case it is interesting to investigate the inner syntagmatic structures of canons with respect to the paradigmatic relations between several canons. The suggestive term 'canon modulation' shall in fact refer to structural analogies in harmony. A typical modulatory effect in harmony is forced by the re-interpretation of a chord in a new harmonic role.
This works with canons as follows:
Consider a canon consisting of 6 exemplars R + s of the inner rhythm R = (0 1 5 6 12 25 29 36 42 48 49 53) with starting points s in the outer rhythm S = (0 22 38 40 54 56). These starting points s in S parameterise the rhythmic roles of the 6 copies of R within the canon. To modulate into another canon within the same translation class means to modulate into a canon with the same fundamental rhythm R, but S replaced by S + t for some t. Candidates for rhythmical re-interpretation are hence the elements in the intersection of S and S + t.
For t = 16 one has S1 = S + 16 = (16 38 54 56 70 0), i.e. there are 4 points in S that can be re-interpreted within S1. This example (listen to the attached midifile ModulationMod72.mid) is constructed as a cyclic sequence of 9 modulations each being a translation of interval 16 mod 72. The re-interpretation of the common rhythm is stimulated by a re-instrumentation of the percussion instruments according to their rhythmic roles.

In this paper we will also focus on another, deeper analogy with harmonic modulation, this is the three-step modulation scheme of Arnold Schoenberg, that has later been modelled mathematically by Guerino Mazzola (Mazzola, 1990), where one not only looks for common chords, but actually studies the process of destabilisation of the old key and the stabilisation of the new one on the basis of an interaction between the two. Finally, we will refer to finite poly-canonic rhythms (with variable periods) for which there exist canons in different cycles.

Augmented rhythmic canons

While we have been concerned only with translations so far, we will now present a suitable generalisation, that leads to canons with augmented voices.
Consider two sequences, R and S, of (invertible affine) symmetries [a,t](x) := a x + t modulo m*n, with R having m entries and S having n. We call R the inner symmetries and S the outer symmetries. Each symmetry consists of an augmentation with factor a and a translation with summand t.
Let A(R) and A(S) denote the augmentation factors of the symmetries in R and
S, and let T(R) and T(S) denote the corresponding translations respectively.
By R*S we denote the mxn - matrix of all pairwise concatenations r*s where r runs through R and s runs through S. This matrix is a candidate of what we call a "canon of symmetries": By |R*S| we denote the set consisting of all entries in the matrix R*S. Now, the pair (R,S) is said to generate the symmetry canon R*S, if T(|R*S|) = Z/m*nZ, i.e. if every translation mod n*m occurs exactly once among the symmetries in |R*S|.
The non-augmented case is characterised through A(R) = (1 ... 1) and A(S) = (1... 1). Moreover, there is a duality of canons in the sense that S*R is the transposed matrix of R*S.
In the augmented situation, however, if we are given a symmetry-canon-generating pair (R,S), it is generally not the case that the pair (S,R) is also canon-generating. In case it is, one has
T(|R*S|) = T(|S*R|) = Z/m*nZ,
but this does not imply |R*S| = |S*R| nor the even stronger condition, that R*S is the transposed matrix (S*R)^ of S*R.
From the musical point of view, it is very interesting to make use of the non-commutativity of symmetries, i.e. to benefit from R*S being different from S*R^.
We say, that a symmetry-canon-generating pair (R,S) has a dual one, if |R*S| = |S*R|.

The following is a suggestive example in the case n = 4 and m = 3:
R = ([11, 0] [5, 1] [5, 3] [11 10]) and S = ([11, 0] [5, 1] [5, 5])

In this example one has:

```
R*S = (
        ([1, 0]   [7, 11]  [7, 7])
        ([7, 1]   [1, 6]   [1, 2])
        ([7, 3]   [1, 8]   [1, 4])
        ([1, 10]  [7, 9]   [7, 5])
      )

(S*R)^ = (
        ([1, 0]   [7, 1] [7, 5] )
        ([7, 11]  [1, 6] [1, 10])
        ([7, 9]   [1, 4] [1, 8] )
        ([1, 2]   [7, 3] [7, 7] )
      )
```

That is, |R*S| = |S*R|, but R*S and (S*R)^ differ from each other in 10 of 12 entries.

Now we explain, how to obtain augmented canons from a symmetry canon. Apply
the entries of R to some onset x mod n*m, say x = 0, in order to generate a inner voice. In the example, R*x = R*0 = (0 1 3 10). From R*x one generates the voices of the desired canon by applying the entries of S to R*x. In the example, we obtain the augmented canon:

S*R*0 = ((0 11 9 2)
(1 6 4 3)
(5 10 8 7))

Analogously, the dual canon turns out to be

R*S*0 = ((0 11 7)
(1 6 2)
(3 8 4)
(10 9 5))

The composer Tom Johnson experimented with the idea to augment the voices in the sense of playing them at different 'tempos'. (Johnson, 2001).
In that case, the lowest common multiple of A(S) defines a long cycle of repetitions of the inner m*n cycle that has to be filled by a suitable number of copies of each voice.

In our example we start with 3 longer voices modulo 4*3*5*11 = 660:
V1 = (0 11 33 110 132 143 165 242 ... 528 539 561 638)
V2 = (1 6 16 51 61 66 76 111 ... 601 606 616 651)
V3 = (5 10 20 55 65 70 80 115 ... 605 610 620 655)

In order to fill the whole cycle mod 660 one needs 11 copies of V1, 5 copies of V2 as well as 5 copies of V3.
Hence, the whole augmented canon consists of 21 voices:
(V1, V1 + 12, ..., V1 + 120, V2, V2 + 12, ..., V1 + 48, V3, V3 + 12, ..., V3 + 48) everything modulo 660. Similarly, its dual canon can be realised with 32 voices. Listen to the attached Midifiles CanonMod12.mid and DualCanonMod12.mid where each Voice is played on a separate pitch of a diatonic scale.

Conclusions

The problem of constructing rhythmic canons tiling the space and the effective possibility to solve it by means of a group-theoretical algorithm shows the usefulness of an algebraic-oriented approach to the formalisation of musical structures. The OpenMusic library OMCanons allows the graphical manipulation of rhythmic operations leading to the complete description of two main families of canons that we tried to present in a formal way: the RCMC-canons and the augmented canons. It also shows how to deal with complex musical transformations, as the modulations between different canons. There are suitable OM-Patches (visual programs) in order to produce all examples presented in this paper and to generalise them according with compositional applications or music-theoretical investigations. Positive reactions of composers already working in different compositional projects suggest looking for other musical transformations (like generalised symmetries), including an extension of the model in the pitch domain. This will be part of a more general OpenMusic library called Zn that is entirely based on the algebraic properties of (cyclic) groups and their application to music.

Bibliography:

Babbitt, M.: 'Twelve-Tone Invariants as Compositional Determinants', The Musical Quaterly 46, pp.246-259, 1960.

Balzano, G.: 'The group-theoretic description of 12-fold and microtonal pitch systems', Computer Music Journal, 4, pp.66-84, 1980.

Chemillier, M.: Structure et Méthode algébriques en informatique musicale (thèse), L.I.T.P., Institut Blaise Pascal, 1990.

Fripertinger, H. : Enumeration in Musical Theory, Dissertation, Graz, 1991.

Johnson, T.: Found Mathematical Objects. http://www.entretemps.asso.fr/Seminaire/Johnson/index.html, 2001.

Mazzola, G. : Geometrie der Toene, Birkhäuser, Basel 1990. (English enlarged version as Topos of Music, forthcoming).

Vieru, A.: Cartea modurilor, 1 (Le livre des modes, 1), Ed. muzicala, Bucarest, 1980 (Revised ed.: The book of modes, Editura Muzicala, Bucarest, 1993).

Vuza, D.T.: 'Aspects mathématiques dans la théorie modale d'Anatol Vieru', Parts 1-4, Revue Roumaine de Mathématiques Pures et Appliquées, 27 (1982), n°2 et 10; 28 (1983), n°7 et 8.

Vuza, D.T.: 'Supplementary Sets - Theory and Algorithm', Muzica, 1, pp.75-99, 1995.

Xenakis, I.: Formalized Music, Indiana University Press, 1971 (Revised Edition: Pendragon Press, 1992).

Supporting Creative Composition: the FrameWorks Approach

Polfreman, Richard
Music Department, University of Hertfordshire.
College Lane, Hatfield, Herts. AL10 9AB.
r.p.polfreman@herts.ac.uk

Abstract:

We present a new system for music composition using structured sequences. FrameWorks has been developed on the basis of Task Analysis research studying composition processes and other user-centred design techniques. While the program only uses MIDI information, it can be seen as a 'proof of concept' for ideas generally applicable to the specification and manipulation of other music control data, be it raw audio, music notation or synthesis parameters. While this first implementation illustrates the basic premise, it already provides composers with an interesting and simple to use environment for exploring and testing musical ideas. Future research will develop the concept, in particular to enhance the scalability of the system.

Keywords:

Music Composition Environments, MIDI Sequencing, Midishare, Musical Structures, Java Applications.

1 Introduction

FrameWorks has been developed as a part of on-going research at the University of Hertfordshire aimed at providing composers with innovative music composition tools that do not require them to become experts in signal processing and/or computer programming (Polfreman and Sapsford, 1995). A major part of this research involved Task Analysis (TA) studies and the development of a Generic Task Model (Johnson, 1992) describing the music composition process. The GTM has been described elsewhere (Polfreman, 1997) and it has been used in the development of Modalys-ER (Polfreman, 1999) as well as FrameWorks (Polfreman, 2001). While Modalys-ER focused on approaches to sound synthesis using physical models, FrameWorks concentrates on ideas of musical structure and reducing *viscosity* and *premature commitment* (Green, 1989) in the system, while doing this in a way that is both simple to use and does not require a mathematical or programming approach. Viscosity refers to the resistance to change of an artefact; in particular whether local changes require many other changes to be made manually elsewhere. Premature commitment refers to the case where key decisions have to be made too early in the process, rather than left to a point where the composer is ready to make them. A bizarre example would be a system that required the composer to specify at the outset exactly the number of notes that a piece was going to contain. While existing systems may be seen as having low viscosity and little premature commitment (Black-

well, Green and Nunn, 2000), we argue that such analysis fails to take into account the complex internal relationships often present in musical work. These relationships, for example, can require many changes to be made throughout a piece in response to a small local change, in order to preserve the integrity of the composition. In most software packages (i.e. not algorithmic composition systems) such changes are not propagated automatically by the system and so a high viscosity can be present. Typically composers are also committed to developing material into a musical structure or form rather than being allowed to start from higher level concerns and then input material.

FrameWorks itself is an early implementation of the ideas emerging from the TA work and it is planned to extend the system in various ways in future work. In its current state it does allow composers to do much that is difficult to achieve with typical software sequencers and encourages composers to experiment with different musical ideas. That said, significant extensions to the current software are needed to produce a truly effective implementation of the concept. FrameWorks is a Java application (1.1.5 or later + Swing/JFC) which uses Grame's free MidiShare system. A free preview release that runs on Windows and Mac OS is available from the University of Hertfordshire from April 2001. A Linux version should be available in the near future.

2 GTM Involvement

The GTM identified three main (interrelated) areas of task performance:

- *Design framework* which involves the setting out of what can be seen as a set of (musical and practical) constraints within which the piece will be composed. This involves defining instrumentation, selecting tools, developing structural ideas and devising systems for creating musical material.

- *Research* which may be necessary before a work can be completed, including many different topics that may be of interest to composers.

- *Produce music* which involves creating the music itself and setting it down in an external form so that it can be performed or tested. This involves the creation of the final deliverable product of the composition process. This can be seen as the application of ideas developed in the first two areas.

In many systems this last task has usually been over-emphasised in typical sequencing tools at the cost of *research* and *design framework* areas. In other cases, in order to achieve more sophisticated musical structures and generation of material, the systems generally require a programming approach to composition, rather than a more intuitive musical one. The GTM analyses these areas further and the structure of FrameWorks reflects at least some of these GTM components. In particular, FrameWorks allows the composer to experiment with some structural aspects of musical composition without using computer programming languages or mathematical constructs. In future development, further parts of the GTM will be implemented in the system in order to support these areas more fully and effectively.

3 FrameWorks

3.1 Overview

FrameWorks is based on a three level structure: *workbench*, *framework* and *sequence*. The workbench is a general area for placing information items relating to a particular composition. This includes both musical and non-musical objects. It is seen as a research supporting area, as the products of research that a composer has undertaken may be used here. The framework is where a musical work is actually put together. A composition is defined in terms of *components* and *relations*. Components are simply containers for musical material. These are placed on tracks (where tracks have a similar role as in typical MIDI sequencers) and can be interconnected via relations. These dynamically maintain relationships between the material in source components and transformed copies of that material in destination components. The sequence level displays the composition without any structural information on a track-based display similar to a traditional software sequencer. This is not for editing but merely to give the composer a flattened view of the completed musical content.

3.2 Workbench

In the current implementation there are five types of *element* that can be placed on the workbench. These are: texts, pictures, diagrams, components and relations:

- Text: These are sections of (currently plain) text. These might be notes taken by the composer relating to musical ideas, inspirational material (e.g. quotes from poetry/prose), texts to be set to music, reminders, etc. These can be created and edited in FrameWorks.

- Pictures: These are picture files (currently .gif) that can be imported into FrameWorks, which then saves its own copy. Again these could be inspirational images, or perhaps pictures of scores or graphics to be used in order to guide some musical aspect of the work.

- Diagrams: These are diagrams of a form similar to those that can be made using packages such as AppleWorks® or Adobe Illustrator®, but less sophisticated. A simple editor is contained within FrameWorks for creating and manipulating diagrams. These might be sketches of the overall shape of a piece or a planned timeline, for example.

- Components: These are sections of musical material (currently just MIDI note events), having a duration, start point and MIDI channel, and are created and edited using a simple piano roll style notation.

- Relations: These express musical relationships between components and invoke transformations such as transposition, inversion, time manipulations and filtering.

Each element on the workbench has a drop down preview of its contents and can be opened (via double-click) to display its contents fully in a new window where the contents can be edited (apart from picture elements). Figure 1 shows an example workbench.

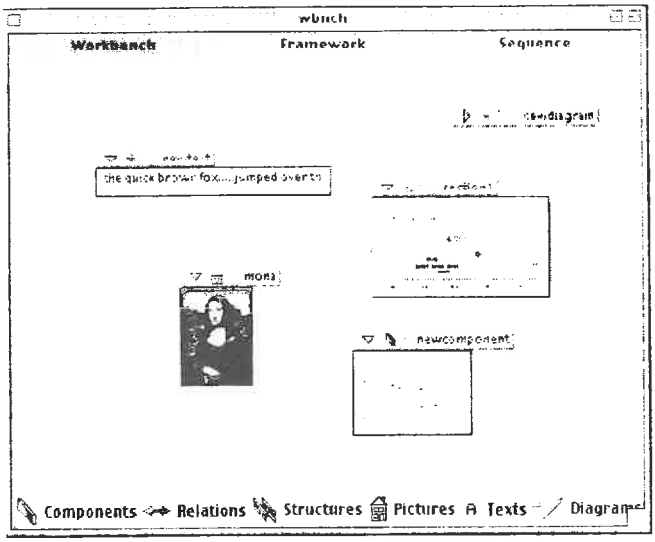


Figure 1: A workbench.

3.3 Framework

In the framework, components are placed on tracks and can be interconnected via relations. There are several types of relation currently used in FrameWorks and we aim to extend this range in future development. At the moment there are four main types:

- Time Relations: These represent a set of time based transformations utilising an arbitrary number of *time maps*. Each map delineates a segment of the source component (with start and end points expressed as percentages of the source duration) which can be played forwards or backwards with a user specifiable speed factor (1.0 = original speed, 0.5 = half speed, 2.0 = twice speed, etc). A time relation comprises a series of time maps applied successively to the source material. These relations can be used simply for time stretching/compressing or to deconstruct and reassemble components in complicated ways.

- Value Relations: These represent time varying transformations of parameters specifying the musical material. In the preview release these only apply to pitch and velocity values in the source material. By using a simple envelope that is stretched to fit the extent of the source material, the composer can shift, scale or invert these source values.

- Filter Relations: These relations pass only some of the source material onto the destination component. They filter by pitch, velocity and duration, with independent control of each.

- Multi-Relations: These are simply combinations of other defined relations into a single relation. An arbitrary number of time, value, filter and multi relations can be used.

More sophisticated relations will be provided as the system develops, but even with these few types, interesting musical ideas can be explored. These initial types have been chosen since they are generally applicable (at least conceptually) to most forms of musical control data, whereas other types are likely to be dependent upon the type of information being han-

dled. For example event based relations (such as retrograde) could be applied to MIDI note data, but would not naturally be applicable to continuous control information.

Components themselves can be manually stretched and compressed to any duration (within the limits of the program), which correspondingly stretches and compresses the material contained within them (and their dependent components). Figure 2 shows an example framework.

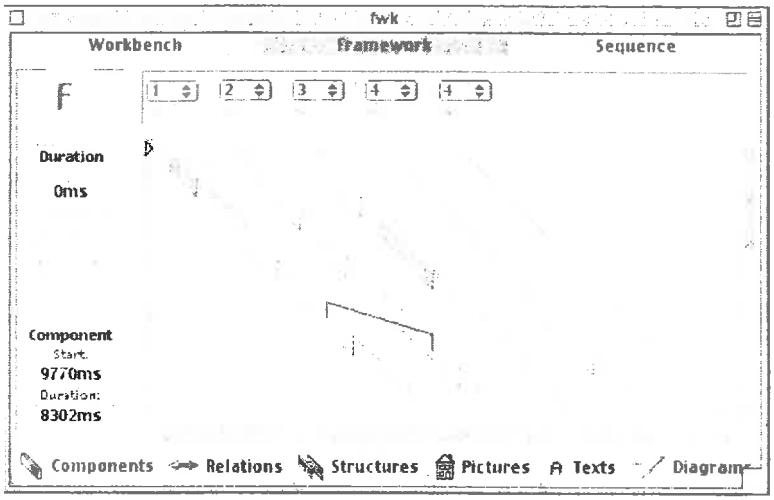


Figure 2: A component-relation framework.

In the framework the left hand area contains the transport controls, time information displays and access to change the duration of the piece. The central area contains the framework with components as parallelogram shapes suspended on diagonal tracks (time flowing from top-left to bottom-right). Relations are indicated by coloured lines between components. At the top of the framework are the controls for each track: MIDI channel/port, mute, solo etc.

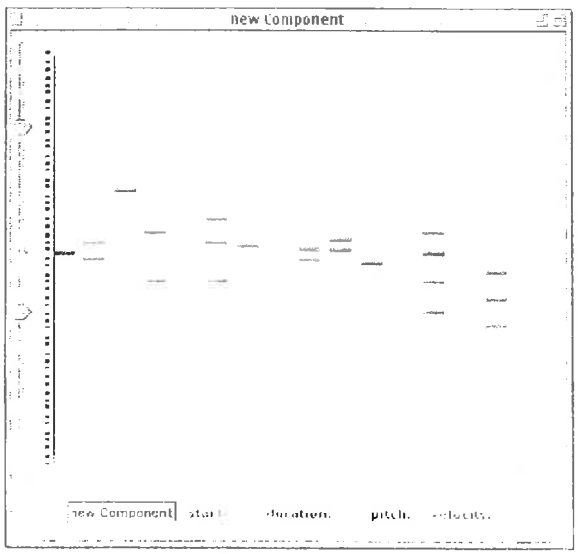


Figure 3: A component editor.

Components are edited via a simple proportional piano-roll style notation (Figure 3). Inserted notes use the velocity and duration set by the sliders on the left of the editor. Notes can be dragged to change pitch and time, the ends dragged to change duration and option-dragged up and down to change velocity. While this editor is basic it does have some useful features. One is the variable time grid to which notes can be 'snapped'. This grid can be any whole number of subdivisions from 1 to 64. By using different time grids with the snap option on, it is very easy to create complex rhythmic patterns of, say, fours against sevens against thirteens.

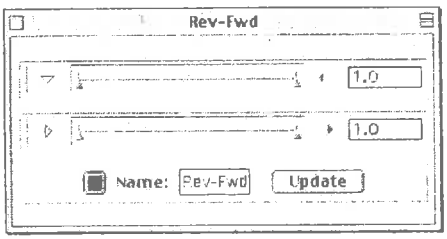


Figure 4: A time relation editor.

Relations are modified using various editors. Figure 4 shows a time relation editor. These have a series of time maps that can be rearranged into any order. Each map has a time line with draggable markers for specifying the segment of material to capture, a direction button and a numerical speed factor. In the figure, the relation simply plays the whole source material backwards then forwards at the original speed. Clicking the expander button on the bottom time map adds another time map to the relation, and any number can be added in this way.

Figure 5 shows a value relation. The editor uses a breakpoint envelope for specifying the value modification over time. A popup menu is provided for choosing the type (shift, scale or inversion) and a button selects between pitch and velocity as the parameter to control.

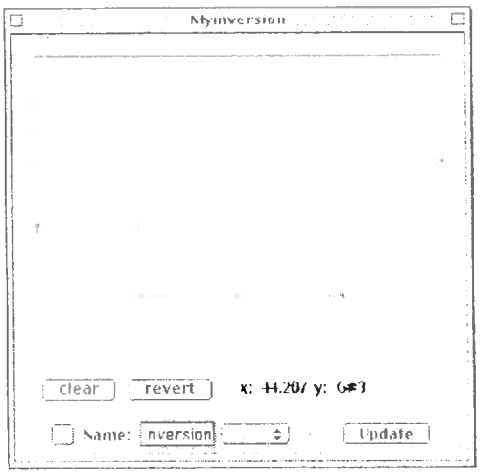


Figure 5: A value relation editor.

Figure 6 shows a filter relation editor. This simply has three bar sliders that allow the user to select a range for each parameter to pass: pitch, velocity and duration. In order to pass non-contiguous ranges, multiple filters can be applied using a multi relation.

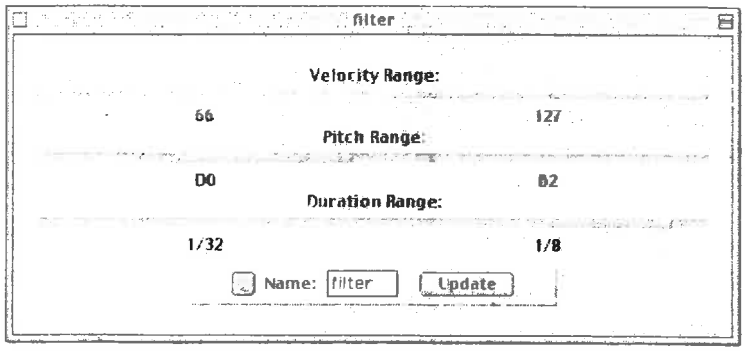


Figure 6: A filter relation editor.

Figure 7 shows a multi relation editor. It can contain as many relations as desired by the user. It is similar in appearance and behaviour to the time relation editor, but each row represents a relation rather than a time map. The relations are applied successively from top to bottom.

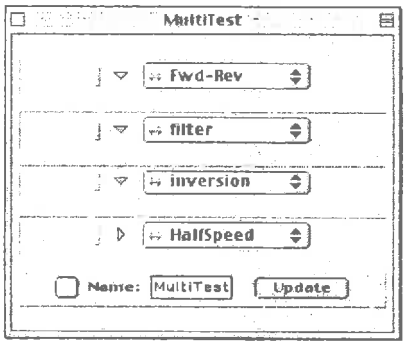


Figure 7: A multi relation editor.

3.4 Sequence

The sequence layer is not intended as an editing level, but purely as an aid to visualise a complete composition without the structural information carried by components and relations.

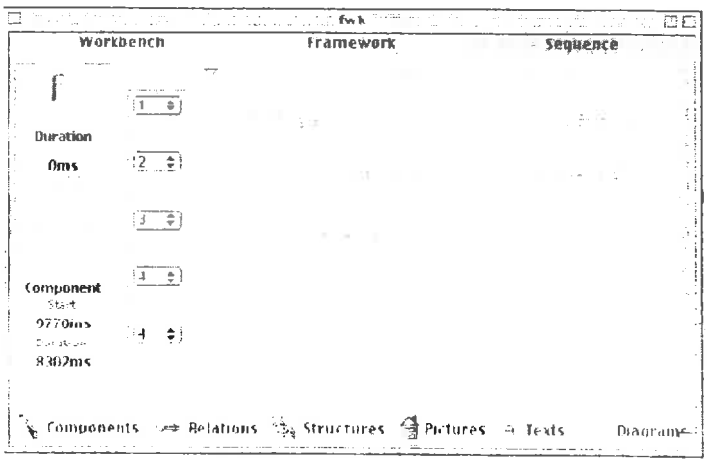


Figure 8: A sequence.

It uses a simple piano-roll type display of the framework tracks. However, the composer can add and remove tracks here as well as change performance parameters such as track MIDI channel/port, mute and solo, and activate the performance controls - play, pause, rewind, etc.

3.5 Stores

Accessible from all levels are the *stores*. These are simply points of quick access to directories containing different element types associated with the current workbench. The idea is that a workbench or framework does not have to have all its associated elements loaded at any one time. A workbench has the file structure indicated in Figure 9.

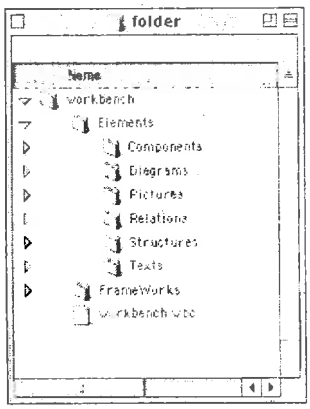


Figure 9: Workbench directory structure.

The workbench is stored in a directory containing the file itself, a sub-directory of frameworks (only one of which can be loaded at a time) and a sub-directory of elements, which in turn has sub-directories for each element type – these are the stores directories. A framework itself saves all of its components and relations independently of the workbench. Currently *structures* are not implemented as an element type, but the idea is that these would be frameworks without any events in their components – a kind of empty template for a part or the whole of a piece. Clicking a stores button (these are along the bottom of the main window) opens a window for accessing the files in that particular store, as shown in Figure 10.

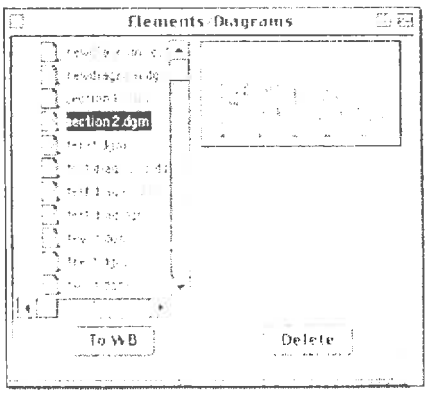


Figure 10: A (diagram) store.

To the left in the Figure can be seen the list of items in the store. Selecting one will show a preview on the right hand side. At the bottom the user can delete the file, add it to the workbench, or if it is a relation or component, add a *copy* of it to the framework. Components and relations can also be copied between the workbench and framework. Deleting elements from the workbench does not delete their corresponding files, but these can be deleted using the stores window.

4 Musical Examples

In this section we demonstrate using simple examples some of the ways in which the framework area of the software can be effective in exploring musical ideas.

4.1 Clapping Music

We start by implementing an abbreviated version (i.e. without repeats) of Steve Reich's Clapping Music. This is a simple composition written for two performers clapping. The entire piece is based around a simple rhythmic pattern – ta-ta-ta_ta-ta_ta_ta-ta_. One part simply repeats this pattern from start to end. The second part cycles this pattern using the iterative application of a rule that takes the first beat of the phrase and moves it to the end each time. This part cycles all the way round until arriving back at the original pattern and plays again in unison with the stationary part. We can implement this relatively easily in FrameWorks. First we define a component that contains this rhythmic pattern (Figure 11).

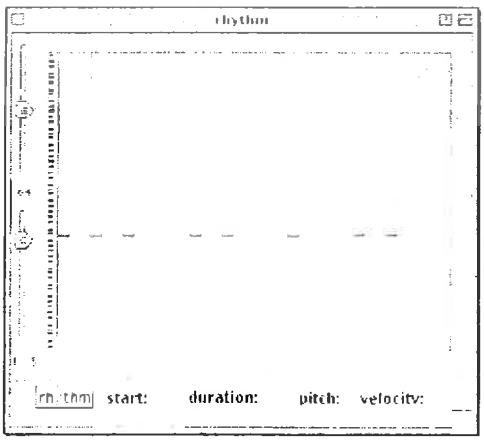


Figure 11: 'Theme' from Clapping Music.

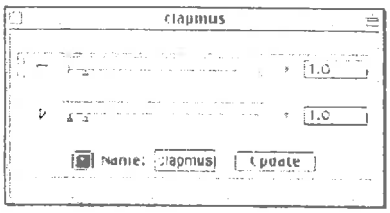


Figure 12: The relation for Clapping Music.

We then define a time relation that carries out the transformation, playing the last eleven twelfths first and the first twelfth last. This is shown in Figure 12. We can then build up a framework using this single source component and the clapping music relation (plus a built-in identity relation).

In the framework (Figure 13), the top left component which is darker than the rest is the source pattern, which is connected to several following components by identity relations (the black links) thus creating the static part. It is then connected across to the first component in the adjacent part with another identity relation. This track then applies the time relation successively in order to create the moving part (actually red links indicating this relation).

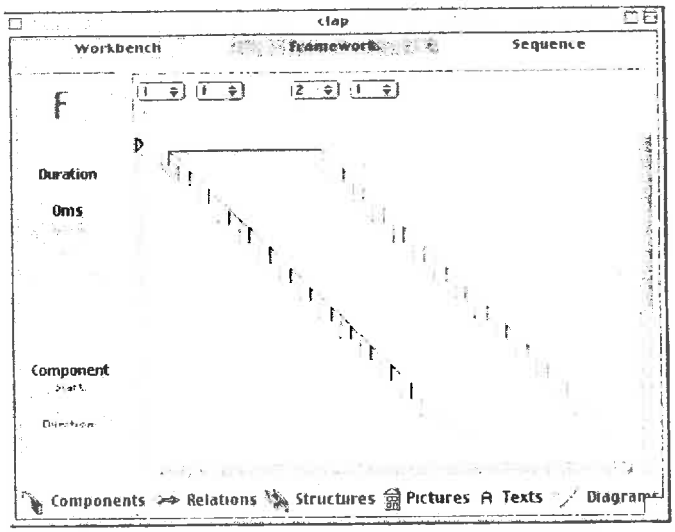


Figure 13: Clapping Music framework.

4.2 Extensions to Clapping Music

Having created a framework for Clapping Music we can extend the composition in simple ways. A first change to make could be to replace the source material. We can preserve or change the rhythmic pattern or assuming a tuned instrument, rather than a simple clap, we can use pitched relationships in the source. Here we try using a theme from another Steve Reich piece, Piano Phase, as the source material (which also uses a pattern in twelve beats) to create a new work - 'Piano Music' perhaps. Figure 14 shows the new source contents, and Figure 15 shows a part of the sequence produced.

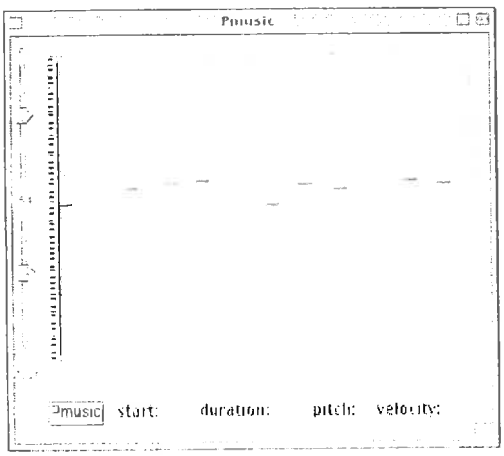


Figure 14: Piano Phase 'theme'.

Figure 15: Start of the sequence 'Piano Music'.

Next we take this piece and add a new relation that rotates the phrase in the opposite sense, i.e. moves beats from the end to the beginning rather than vice versa. This relation is shown in Figure 16.

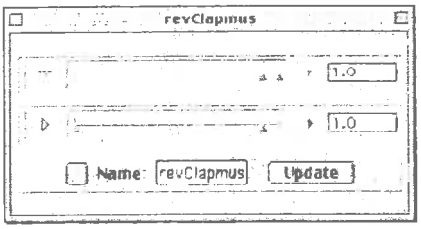


Figure 16: Reverse clapping music relation.

We then add a new track that uses this relation to create a third part that cycles in the reverse direction to the original moving part, as shown in the framework in Figure 17.

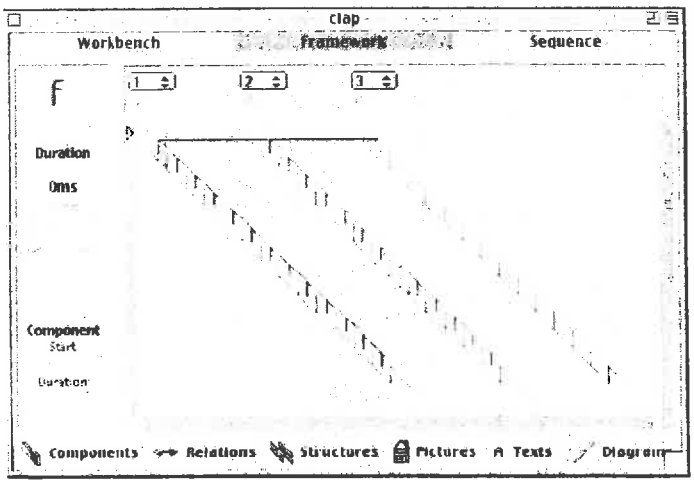


Figure 17: Framework with anti-phase moving parts.

Finally we again replace the source material, here with an original 'theme', shown in Figure 18, producing a sequence with the opening shown in Figure 19. We can of course experiment by editing this material freely and immediately being able to hear the results on the entire piece without having to edit any other information. Thus we can explore different possibilities within the same musical structure very easily, as well as experiment with different structures using the same material.

Clearly there are many other extensions we could make using other relation types, modifying the structure in terms of the number of components and where these are located, etc. These examples serve to show the fluidity of the system in terms of modifying entire compositions without necessarily making many changes to the framework.

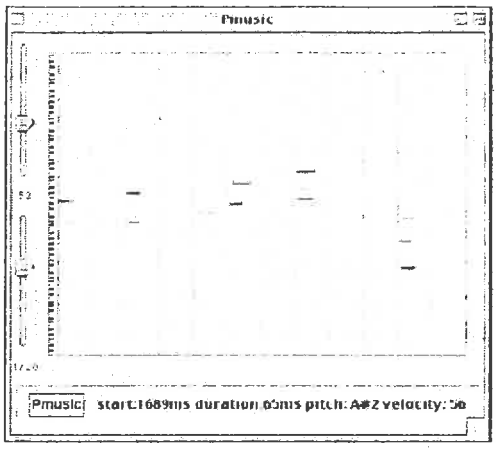


Figure 18: Final 'theme'.

Figure 19: Opening of piece using the final theme.

5 Conclusions & Further Research

FrameWorks is an early prototype system developed in order to try out our design ideas and allow composers to test them in practice. FrameWorks has currently several limitations:

- Flat Structures: There is no support for nesting components within other components to create hierarchical musical structures. This is necessary in order to handle large-scale musical works and allows even short works to be represented more effectively. Graphical presentation of the frameworks with added hierarchy would be difficult in a two-dimensional system and so we will be exploring the use of three-dimensional notations in the system.

- Data Types: FrameWorks only supports MIDI note events and should be extended to allow for MIDI controller and other data. In the longer term we hope to develop an open system that can be used with a variety of data types (e.g. audio, synthesis parameters), preferably via a plugin architecture. The use of the Java Sound API rather than Midishare may be useful in developing for a wider range of hardware platforms.

- Notations: The notation systems used are quite primitive and need to be enhanced with possible multiple notation options.

- Relations: These are limited to a few key types and should be extended to other transformations. An important enhancement would be to make the system aware of musical keys so that transformations can conform to key signatures rather than being absolute. Input from composers and perhaps analysis of extant works will be important in defining these. Again it would be desirable to use a plugin architecture for these.

- Generation Systems: Currently the composer can only enter events into components by recording or manual insertion of events. While the system is not aimed at being a true algorithmic composition environment it would be useful to add some support for other means of generating musical events through processes and constraints (where processes are methods of value selection and constraints govern what values can be selected).

- General: There are some general useful features lacking in the preview release that we hope to add before version 1.0 Final is made available, particularly in terms of full editing support and on-line help one expects in modern software products. Feedback from users of the preview release will be taken into account in the development process. There are also some efficiency issues noticeable with complex frameworks, some rounding problems (due to a millisecond time resolution) and other current bugs.

In conclusion, we believe that FrameWorks even in its current form provides a novel system for music composition that offers composers interesting ways of working that in some ways better reflect the conceptual levels used by composers in composition tasks. While the examples used here were simple and lent themselves readily to the FrameWorks model, more sophisticated and less process driven music can also be created effectively within the system, although hierarchical arrangements are yet to be supported. We hope to gain substantial feedback from composers regarding their use of the system and use their needs as a guide to future development. As the system develops in sophistication we hope to use it in the analysis of extant works and believe that it will also provide a useful pedagogical tool for music educators.

Acknowledgements

My thanks go to Stephane Letz at Grame for help with MidiShare/Java in the development of FrameWorks.

References

Blackwell, A.F, Green, T.R.G. and Nunn, D.J.E. 2000. Cognitive Dimensions and Musical Notation Systems. *Workshop on Notation and Music Information Retrieval in the Computer Age.* At the 2000 ICMC, Berlin.

Green, T. R. G. 1989. Cognitive dimensions of notations. *People and Computers V, Proceedings of the HCI '89 Conference.* 443-460 Eds. A. Suttcliffe and L. Macauley, CUP.

Johnson, P. 1992. *Human computer interaction: psychology, task analysis and software engineering.* McGraw-Hill.

Polfreman, R. and Sapsford, J. 1995. A human factors approach to computer music systems user-interface design. In *Proceedings of the 1995 ICMC.* ICMA.

Polfreman, R. 1997. *User-interface design for software based sound synthesis systems* (PhD Thesis). University of Hertfordshire.

Polfreman, R. 1999. A task analysis of music composition and its application to the development of Modalyser. *Organised Sound*, 4(1):31-43, CUP.

Polfreman, R. 2001. *FrameWorks User Documentation.* University of Hertfordshire.

Automatic modeling of musical style

Assayag Gérard (1), Bejerano Gill (2), Dubnov Shlomo (2), Lartillot Olivier (1)
(1) Ircam, Centre Georges-Pompidou
1, place Igor-Stravinsky
75004 Paris
prénom.nom@ircam.fr
(2) Université Ben Gurion

Abstract : In this paper, we describe and compare two methods for unsupervised learning of musical style, both of which perform analyses of musical sequences and then compute a model from which new interpretations / improvisations close to the original's style can be generated. In both cases, an important part of the musical structure is captured, including rhythm, melodic contour, and polyphonic relationships. The first method is a drastic improvement of the Incremental Parsing (IP) method, a method derived from compression theory and proven useful in the musical domain. The second one is an application to music of Prediction Suffix Trees (PST), a learning technique initially developed for statistical modeling of complex sequences with applications in linguistics and biology.

Keywords : Unsupervised learning, Musical style, Compression, Predition Suffix Tree (PST), Probabilistic Finite Automata (PSA), Lempel-Ziv (LZ), Stochastic, Quantization, Constraints, Loop, Redundancy, Musical Parameters, Mar\-kov predictor.

Style Modeling

By Style Modeling, we imply building a computational representation of the musical surface that captures important stylistic features hidden in the way patterns of rhythm , melody, harmony and polyphonic relationships are interleaved and recombined in a redundant fashion. Such a model makes it possible to generate new instances of musical sequences that respect this explicited style. It is therefore an analysis by synthesis scheme, where the closeness of the synthetic to the original may be evaluated and may validate the analysis. Our approach is unsupervised, that is we want an automatic learning process that may be run on huge quantities of musical data. Interesting applications include style characterization tools for the musicologist, generation of stylistic meta-data for intelligent retrieval in musical data bases, convincing music generation for web and game applications, machine improvisation with human performers, computer assisted composition.

The IP method

The Incremental Parsing algorithm is inspired by the analysis part of compression techniques of the Lempel-Ziv family. To understand how an idea derived from the compression field might be useful for our purpose, it is important to see that, as has been stated by several authors, compressing is equivalent to understanding, because in order to encode efficiently incoming information one has to perform a fine analysis of the way redundancy is organized. compression algorithm can be split into two phases: first, it reads the input sequence and constructs a model that captures redundancy, and then it generates the compressed code of

this sequence with respect to the model. In our case, the second phase is replaced by a stochastic navigation through the model in order to generate new sequences.

During the generation process, a context-inference scheme is applied. The sequence formed by recently generated objects (a particular suffix of this sequence) is the context, from which a prediction on the next object to come is made with regard to a contextual probability distribution. So, the analysis part must provide a dictionary of such possible contexts along with their possible continuations.

First, the dictionary is empty and IP incrementally reads the sequence. At each cycle, it selects a pattern, from the current position to a further position, such that this pattern is the shortest one which is not already in the dictionary. Every left prefix of this pattern may become a context, and every object that follows this prefix may become a continuation. It is easy to see that an optimal representation (in space) for such a dictionary is a prefix tree where a branch descending from the root to any node is a context, the childs of any nodes are the continuation objects to the context going from root to this node, and where the cardinalities of the subtrees at a certain node encodes the probability distribution for the objects at the the root of each subtree. An optimal representation in generation time is a suffix tree where the context are reversed (i.e. from a leaf to the root) and continuations stored as pointers associated to every node.

We have loosely used the term *sequence* as an ordered list of objects. In order to capture a significant amount of musical substance, we shall, in a pre-analytic phase, cut the musical data (generally in Midi format) into slices which beginning and end are determined by the appearance of new events and the extinction of past events. Every slice has a duration information, and contains a series of channels, each of which contains pitch and velocity information and whatever available musical parameters. These slices are serialized in sequences submitted to analysis (these slices will be referred to by the word *object* or *symbol*, and the set of possible symbols as *alphabet*).

We have described here the basic IP algorithm. This algorithm had already been tested with interesting results, but had certain drawbacks that made it quite impractical in specific situations. The improvements presented here are divided into four sections: pre-analytic simplification, generative constraints, loop escape, analysis-synthesis parameter distribution.

Pre-analytic simplification

Real musical sequences - for example MIDI files of a piece interpreted by a musician - feature fluctuations of note onsets, durations and velocities, inducing a complexity which fools the analysis: the alphabet size tends to grow in an intractable way, leading to unexpected failures, and poor generalisation power. We have thus developed a toolkit containing five simplification filters:

- the *arpeggio filter* vertically aligns notes which are attacked nearly at the same time,
- the *legato filter* removes overlap between successive notes
- the *staccato filter* ignores silence between successive notes,
- the *release filter* vertically aligns note releases,
- the *duration filter* quantizes the durations in order to reduces the duration alphabet.

These features can be tweaked by the user, using thresholds (e.g. a threshold of 50ms separates a struck chord on the piano from an intended arpeggiated chord). Using the simplification toolkit, Midifiles containing real performances that were intractable with the basic IP now become manageable, opening new perspectives, because this particular kind of musical data, full of idiosyncrasy, is of great value as a model for synthetic improvisation.

Generative constraints

It is now possible to specify constraints during the synthetic phase. At each synthetic cycle, if the constraint is not respected by the new generated symbol, this generation is canceled and a new symbol is tried. If no symbol is satisfying, the algorithm backtracks to the previous cycle, and more if necessary. One interesting constraint is called the continuity constraint: at any cycle of the synthetic phase, it is possible that no context sequence be a suffix of the already generated sequence. The maximum context is thus the empty context. In these cases, the algorithm generates a continuation symbol of the empty context, that is to say, any symbol, with a stochastic model corresponding to its occurrence in the original sequence. The obtained result is a musical discontinuity. To avoid this, it is possible to specify a continuity constraint which imposes a minimum size of context anytime during the synthetic phase.

Loop escape

The synthetic phase may easily enter into an infinite loop state. Here is one example: at one cycle, the maximal context *A* proposes only one continuation symbol; once this symbol is generated the new maximal context *B* features only one continuation symbol, and the new maximal context is *A*, again. The next contexts will be *B, A, B, A,* ... This tends to happen when the input data contains contiguous repetitions, which is often the case in music. We describe a mechanism to detect and escape these loops.

The previous example is very simple, because this loop was totally deterministic, and had only two states. At one state of a loop, it may possible to consider several possible continuation symbols, but this choice leads, sooner or later, to return back to this present context or a previous one. We introduce the concept of context-generated subtree, which consists of the exhaustive set of all the possible contexts that may be met after the present one. There is a loop if and only if this context-generated subtree contains few contexts. When the size of the context-generated subtree is below a user-specified threshold *N*, an *N*-order-loop is detected. The loop phenomenon is principally due to the fact that the synthetic phase searches for the maximal context, which is unique and proposes few alternative continuation symbols. In the case of a loop, we loosen this constraint and examine not only the maximal context, but also smaller ones, which escape the loop in most cases.

Analysis-synthesis parameter distribution

The analytic phase consists of finding the redundancy inside an original sequence of symbols. The trouble is there may be so much complexity and diversity in musical sequences that the size of the alphabet may be of the same order of the length of the sequence. Therefore, little redundancy would be observable. Moreover, each symbol consists, as said before, of several

channels, each channel consists of notes, and each note consists of different parameters. So a symbol is in fact a Cartesian product of several musical parameters.

In order to increase abstraction and power in the analysis, we allow the system to discard some parameters, for example the velocity. The retained parameters will be called *analysis information*. In this way, it is obviously possible to increase redundancy, because it implicitly organizes the alphabet into equivalence classes: a chord struck two times with different velocities is nevertheless the same chord with the same harmonic function, and it will be detected as such. The problem is, in the synthetic phase, all the discarded information cannot be retrieved. For example, if we choose to discard note durations and dynamics, we finally obtain isorythmic and dynamically flat musical sequences, which sounds like musical box production. A better solution is to store in the model the excluded information.This information is thus called *synthetic information*. The analytic phase is now performed on classes inside the initial symbol set, and during the synthetic phase, synthetic information, e.g. expressivity, may be reconstructed.

This solution has significant advantages. First, generated music regains much diversity, spirit and human appearance of the original one. Moreover, since it is possible to restrict analytic information and therefore find more redundancy in the original sequence, the synthetic phase becomes less constrained. At each synthetic cycle, every context features many more possible continuations than before.

Implementation

The software is implemented as a user library in an Open Source visual programming language developed at Ircam, called *OpenMusic*. Each step of the algorithm - pre-analysis, analysis, synthesis, and post-synthesis, is a function, which, in the musical representation software, is represented by a box featuring inlets and outlets. All parameters may be tuned up by the user. Indeed, synthetic constraints, the distribution of analytic and synthetic information, and the reconstruction of the synthetic information may be explicitly formulated through visual expressions.

Musical experiments

A lot of musical experiments have been carried in order to test the new IP algorithm. Midifiles gathered from several sources, including polyphonic music, piano music, and which style ranges from early music to hard-bop jazz have been submitted to the learning process. Experiments show that the combination of the simplification tool box and the new analysis-synthesis distribution scheme improves dramatically the results in the case of 'live' music, and in cases where the overall complexity leads to a huge alphabet. We show convincing examples, including, a set of piano improvisations in the style of Chick Corea, another one in the style of Chopin Etudes, polyphonic Bach counterpoint, 19th century symphonic music, and modern jazz style improvisations derived from a training set fed by several performers asked to play for the learning system.

PST

Asymptotically it has been shown that IP predictor outperforms a Markov predictor of any fixed finite order. This surprising property of the IP scheme derives from the counting interpretation of the IP procedure. In this interpretation, the IP predictor is viewed as a set of sequential predictors operating on separate bins, where each phrase derived in the process of IP parsing is referred as the bin label. Since IP is unbounded in its length, it can be shown that for any finite k-th order Markov predictor, the long contexts of IP serve as refinements of the k-th order Markov predictors for sufficiently long sequences. Thus, the total number of errors due to a long context prediction is smaller for IP than the error in the case of a limited memory Markov predictor. When calculating the overall error regime, it turns out that asymptotically the longer terms dominate (since the length of the string grows) and eventually the IP scheme outperforms any finite order Markov predictor.

In practice the strings (music sequences) are of a finite order, and moreover, the size of the IP tree is bounded by a small finite size. Due to these limitations it is desirable to consider modeling schemes that might be more optimal for shorter-term situations. Another important feature of IP that seems redundant for our needs is the sequential nature of its operation. In our application, the goal is generation of new sequences that maintain similar statistical properties to the reference source. We use the prediction probabilities as the statistics generator, but we are not bounded by a requirement to rely on the past only. Allowing one to use the whole sequence for estimation of the statistics might help improve the performance.

Ron et al developed a variable length Markov model termed Prediction Suffix Tree (PST). It has been shown that PST is a subclass of Probabilistic Finite Automata called PSAs. PSA is a variant order Markov chain in which the memory is variable and in principle unbounded. Given a finite size PSA, there exists an equivalent PST of a slightly larger size that produces the same probability output. Moreover, an efficient learning algorithm exists that allows one to construct a PST from samples generated by PSA. Now, if we consider the PSA as a common ground relating it to IP, we can consider the similarities and differences between the two approaches. One can see that both methods are similar in terms of their use of a variable length context for determining the probability for next symbol. The basic estimation procedure of the PST though is significantly different from that of IP.

IP/PST comparison

Batch vs. on-line

PST parsing is not on-line in nature, as IP is - the training text is viewed as a whole unit and symbol frequencies are observed over the whole text. Thus there is no *arbitrary parsing* as in LZ where a single symbol change in the sequence may have deep influence on the dictionary structure. So PST may prove more powerful for short sequences but is not practicable for real time improvisation situations, where the on-line nature of IP will be adapted.

Selectivity

while IP goes over the training text and parses all of it, the PST learning algorithm looks at the text as a whole and picks for its dictionary only patterns considered *relevant*. Relevance is defined through several parameters pertaining mainly to the number of times that subsequence has appeared in the whole text, and to how different prediction of the next symbol using that substring differs than that using its suffix (e.g. a PST will learn the suffix/subsequence *BCBA* only if it appears enough times in the text, and the next symbol prediction (*BCBA?*) is different enough than that achieved using only *CBA* (*CBA?*)). This selective parsing can lead to more discontinuities and a smaller repertoire to improvise on, but the context→inference rules stores in the trees may be considered as more motivated than in the case of IP.

From these two remarks, one could say that IP is more adapted in generative applications, especially improvisation and real-time, and PST is more precise and complete, and is a good start for musicology applications where one wants to achieve the finest description.

Other experiments are undergoing and a more precise comparison will appear in the final paper.

Conclusion and new directions

In the present state of researches, we have to consider music as a sequence of symbols. It could be more interesting to analyze directly the bi-dimensional score structure. This would enable an intelligent analysis of a fugue for example.

Although the context-inference approach may be compared with the implication/realisation view of Meyer, we have to acknowledge that his view is more powerful since his expectation is in long term, and not a systemic first order expectation like ours. But we suspect hard computability problems behind the long term approach.

In our work, learning is indeed unsupervised since we do not feed our system with musical knowledge, rather it analyzes music with constrained algorithms and does not proceed to any inductive inference. Another approach would be trying to induce automatically some conclusions about the presence of musical structures, with the help of minimal cognitive mechanisms.

Other statistical techniques have to be experimented in music and compared to the two presented here. PPM for instance, can be thought of as going half-way between IP and PSTs. They are on-line (like IP) but they model *k*-terms of growing *k* with respect to data wealth (like PSTs).

Finally, one may try instead of modeling sequence x1 x2 … xn based on its own redundancies, to try and model x1 x2 … xn based on the redundancies of a second, correlated (eg 2nd voice) sequence y1 y2 … yn (transducer). Then, the generation of a X-type sequence

could be constrained by an incoming Y-type sequence. This could lead to a synchronous improvisation scheme, where the computer, instead of providing 'answers' to a human player, as usual, would play synchronously with him, keeping an overall polyphonic consistency.

Improviser des séquences d'accords de jazz avec des grammaires formelles

Marc Chemillier
Université de Caen
marc@info.unicaen.fr

1. Introduction

Cet article décrit un logiciel implémentant un générateur d'accords dans un environnement de type « DJ mix », d'après les principes de l'harmonie jazz. Le programme produit une sortie midi au moyen de 1) un générateur de séquences d'accords de jazz à base de règles de grammaire, et 2) un dictionnaire de mesures précomposées, associant les symboles d'accords à des échantillons au format midi. L'aspect harmonique d'un accompagnement de jazz repose sur une grille harmonique tonale de *n* accords, répétée en boucle avec des variations autant de fois que nécessaire. Le pianiste improvise une réalisation de la séquence d'accords, en variant les renversements, et en introduisant de nouveaux accords par l'application de substitutions harmoniques à la grille d'origine. Le sujet principal de cet article est l'implémentation de cette technique de réalisation et de substitution harmonique.

2. La grammaire de Steedman

En 1984, Steedman a proposé une grammaire dans le but de décrire les substitutions harmoniques utilisées par les musiciens de jazz. Sa grammaire est exprimée sous la forme de six règles de réécriture prenant comme séquence initiale la grille de base du blues et générant une variante du type de celles jouées par les musiciens de jazz moderne [Steedman 1984]. Il a déduit les règles de l'analyse d'un corpus de huit grilles empruntées à un livre de [Coker 1964]. Dans le prolongement de son article, d'autres auteurs ont apporté de nouveaux développements à son intuition originale [Johnson-Laird 1991], [Pachet 1999], [Chemillier 2001].

L'exemple suivant montre deux états d'une grille de blues en *do* (*C* en chiffrage anglo-saxon). Le premier est la grille de base.

C	C	C	C7
F	F	C	C
G7	G7	C	C

C	C	C	Gm7
F	F	C	C
Dm7	G7	C	C

Le second résulte de deux substitutions appliquées respectivement à la quatrième mesure :

C7 Æ Gm7 C7

et aux neuvième et dixième mesures :

G7 G7 Æ Dm7 G7

La grammaire de Steedman consiste en six règles de ce type, chacune exprimée avec une variable *x* qui désigne l'un quelconque des degrés *C, C#, D,* ... Ainsi, chaque règle de Steedman correspond en réalité à douze règles différentes. Les fonctions *Sdx, Dx, Stbx, Stx, Mx, Lx* sont respectivement la sous-dominante, la dominante, le sus-tonique bémole, la sus-tonique, la médiante, et la sus-dominante de *x*. Dans la règle 3, la variable *w* désigne un accord indéterminé, dont le degré n'a pas été modifié par une règle appliquée précédement, et dont le chiffrage n'est pas 7. Les accolades de la règle 6 indiquent un choix.

Règle 1:	*x Æ x x*
	x7 Æ x x7
	xm7 Æ x xm7
Règle 2:	*x Æ x Sdx*
	x7 Æ x7 Sdx
	xm7 Æ xm7 Sdx
Règle 3a:	*w x7 Æ Dx7 x7*
	w x7 Æ Dxm7 x7
Règle 3b:	*w xm7 Æ Dx7 xm7*
Règle 4:	*Dx7 x7 Æ Stbx7 x7*
	Dx7 x Æ Stbx x
	Dx7 xm7 Æ Stbxm7 xm7
Règle 5:	*x x x Æ x Stxm Mxm*
Règle 6:	*x x {Dx, Stxm7, Lxm7} Æ x x#°7 {Dx, Stxm7, Lxm7}*
	xm xm {Dx, Stxm7, Lxm7} Æ xm x#°7 {Dx, Stxm7, Lxm7}

Tous les accords sont à la fois des symboles terminaux et non-terminaux. Une règle supplémentaire permet de réécrire l'axiome de la grammaire en une séquence correspondant à la grille de base du blues.

3. Parsing automatique avec Lex

Plus récemment, Steedman a publié un autre article consacré au même sujet, pour étudier certaines difficultés qui apparaissent dans l'implémentation d'un analyseur construit à partir de sa grammaire, à cause du fait que la grammaire est context-sensitive [Steedman 1996]. En effet, comme on peut le voir ci-dessus, certaines règles ont plus d'un symbole non terminal du côté gauche (c'est le cas pour la règle *G G7 Æ Dm7 G7* dans l'exemple précédent), et par conséquent ne sont pas de type régulier, ni même de type context-free, dans la classique hiérarchie de Chomsky. Il semble impossible d'implémenter cette grammaire sous forme d'un analyseur avec des outils comme Lex ou Yacc.

Nous avons proposé une solution au problème en formalisant certains aspects de la grammaire qui ne sont pas explicite dans les règles décrites ci-dessus, concernant la durée des accords. Dans la seconde règle de l'exemple précédent, les accords des deux côtés de

la règle ont la même durée. Mais ce n'est pas le cas dans la première règle où les accords du côté droit ont une durée moitié plus courte que l'accord du côté gauche. Aussi l'action de règle de ce type doit-elle être limitée, dans la mesure où la durée des accords dans une grille de jazz n'est généralement pas plus courte qu'un quart de mesure. Pour prendre en compte cette restriction, nous avons introduit une modification dans la forme des règles de Steedman, en ajoutant un nouveau symbole à la grammaire pour représenter les barres de mesure (noté /). Par exemple, la règle

x Æ x x

est remplacée par la règle

/ x / Æ / x x /

ce qui signifie que la règle précédente ne peut être appliquée que dans une case de la grille ne comportant qu'un seul accord. Les nouvelles règles obtenues à partir des règles originales sont désormais capable de contrôler que la durée des accords reste supérieure au quart de la mesure.

On peut prouver facilement que la nouvelle forme de cette grammaire est équivalente à un automate fini. La durée d'un accord étant restreinte à des valeurs supérieures au quart de la mesure, et le nombre de mesures étant fixé, il est évident que le nombre de séquences d'accords possibles générés par la grammaire est fini. Aussi peut-il être engendré par un automate fini, et nous avons construit un algorithme calculant l'automate correspondant.

En utilisant Lex, nous avons implémenté cet automate sous forme d'un analyseur, qui prend en entrée une séquence d'accords, et affiche Reconnu si la séquence peut être engendré par la grammaire, et Non reconnu dans le cas contraire. Pour réaliser cette implémentation, Lex utilise une description de l'automate sous forme d'expression régulière, et produit un programme écrit en C qui implémente l'analyseur. Le code décrivant l'analyseur est reproduit dans [Chemillier 2001]. L'analyseur obtenu prend en entrée une séquence d'accords séparés par une barre de mesure (/), comme dans l'exemple ci-après, dans lequel la séquence d'entrée est le célèbre *Blues for Alice* composé par Charlie Parker.

$ blues
> F / Em7 A7 / Dm7 G7 / Cm7 F7 / Bb7 / Bbm7 Eb7 / Am7 / Abm7 Db7 / Gm7 / C7 / F7 / Gm7 C7
Reconnu

4. Langages formels d'accords et renversements échantillonnés

Notre logiciel générant des accords de séquence de jazz est une combinaison de deux modules. Le premier applique aléatoirement des règles de substitution à la séquence d'accords, engendrant ainsi un langage formel d'accords de type régulier. Le second module choisit des renversements pour les accords successifs de la séquence obtenue, en sélectionnant des renversements échantillonnés dans une base de données. L'ensemble du

programme a été écrit en Lisp dans l'environnement OpenMusic, qui est un langage visuel pour la composition musicale développé à l'Ircam. Des exemples de fichiers midi calculés par le programme sont disponibles sur le Web (www.info.unicaen.fr/~marc/publi/grammaires), et le code Lisp du programme est reproduit dans [Chemillier 2001].

Le module de renversement d'accords transforme les accords en données midi afin de produire une séquence midi qui est jouée par un synthétiseur. Il repose sur une table d'association entre accords et données midi préfabriquées appelées *samples midi*. Cette table n'est pas limitée a des entrées constituées d'un seul accord, mais associe des sample midi a des séquences de n accords consécutifs. Cela signifie que le module lit la séquence d'entrée à travers une fenêtre de longueur n. Ainsi, le modèle formel de ce module est-il une transduction à états finis.

Le module de substitution opère en choissant aléatoirement une position dans la séquence d'accords où une règle peut être appliquée. La procédure de base de ce module prend en argument une grille et un entier k, et retourne une nouvelle grille obtenue après k application du processus de substitution à la grille d'entrée. Voici deux exemples, avec respectivement une et dix substitutions appliquées à la grille de base du blues mémorisée dans la variable *grille-base*. Dans ces exemples, les accords engendrés par le processus de substitution sont marqués avec une étoile.

? (reecriture *grille-base* 1)
((c) / (c) / ((g *) 7) / (c 7) / (f) / (f) / (c) / (c) / (g) / (g 7) / (c) / (c))

? (reecriture *grille-base* 10)
((c) / (c) / (c) / ((c# *) 7) (c 7) / (f) / ((f# *) 7) / ((f *) 7) / ((bb *) 7) / ((eb *)) ((eb *) 7) / ((d *) 7) ((c# *)) / (c) / (c))

Dans l'état actuel du logiciel, l'utilisateur peut interagir en temps réel avec lui, réalisant ainsi une véritable improvisation. Cette interaction peut se faire de trois différentes façons.

Tout d'abord on peut modifier la profondeur du processus de substitution, qui correspond au nombre k de règles appliquées dans l'exemple ci-dessus. Pendant qu'une grille est jouée, l'utilisateur a la possibilité de modifier la valeur de k pour le calcul de la grille suivante. Ainsi, la séquence harmonique peut-elle évoluer entre l'état de base de la grille sans substitution, et un état très modifié comportant de nombreuses substitutions.

La seconde possibilité d'interagir avec le système est d'orienter manuellement le choix des accords. Cette possibilité est étroitement liée à la propriété que nous avons mentionnée dans la section précédente concernant le type régulier du langage de séquences d'accords engendré par la grammaire de Steedman (restreintes à des durées supérieures à la pulsation). L'accord $X_i{+}1$ qui peut suivre un accord donné X_i est le résultat de l'activation de n'importe quelle règle de substitution appliquée à la séquence d'accords courante, mais au-delà, cet accord peut résulter également de l'activation de k règles successives appliquées de manière cumulative. Par exemple, si l'on considère la séquence suivante

$XYXXXZ,$

le successeur de Y est X. Si l'on applique la règle de substitution $XZÆZZ,$ on obtient la variante

$XYXXZZ, \quad k = 1,$

ce qui ne change pas les successeurs possibles de Y. Par contre, si on itére le processus en appliquant une deuxième, puis une troisième fois la règle, on obtient les variantes

$XYXZZZ, \quad k = 2,$
$XYZZZZ, \quad k = 3,$

où les successeurs possibles de Y sont désormais X et Z. Bien que k puisse prendre n'importe quelle valeur entière, il est possible de calculer tous les X_{i+1} pouvant suivre un X_i donné, grâce au fait que la grammaire de Steedman est équivalente à un automate fini. D'une manière usuelle, cet automate peut être représenté par un graphe avec un nombre fini d'états, et des flèches représentant les transitions entre états. Durant le calcul, il est possible d'afficher l'état exact du système et de donner à l'utilisateur la possibilité de choisir quelle transition sera activée à partir de l'état courant.

Une troisième manière d'interagir avec le système est de choisir de nouveaux samples midi pour le module de renversement. La base de données contenant tous les samples midi disponibles est organisée selon différentes tables correspondant à des effets variés, des ambiances contrastées, etc. Dans l'état présent, les renversements stockés dans la base de données sont des arrangements dans le style des musiques électroniques actuelles de type house, ambient (comprenant basses, percussions et bruitages divers). Une table très simple d'arrangements pour piano dans le style boogie-woogie est disponible sur le Web à l'adresse indiquée ci-dessus.

4. DJing avec un générateur d'accords

Cette manière d'interagir avec le système en choisissant des samples midi correspondant à différents types de renversements rappelle la technique des DJ consistant à choisir des samples dans des disques et à les mixer les uns avec les autres. La tâche du DJ n'est autre que la sélection d'extraits de musique précomposée (stockée sur des vinyles), et la combinaison de ces extraits à l'intérieur d'une nouvelle composition (réalisée avec deux platines connectées à un mixeur). Il existe de nombreux logiciels simulant la tâche du DJ en permettant le mixage en temps réel de différents types de fichiers audio (wav, aiff, mp3). Notre système partage quelques aspects en commun avec ce type de logiciel. La principale différence réside dans le fait que les samples ne sont pas audio mais midi, et surtout qu'ils ne sont pas choisis manuellement, mais à l'aide d'une procédure automatique implémentant la grammaire de substitution décrite ci-dessus. Aussi notre générateur d'accords peut-il être considéré comme une extension des traditionnels logiciels de DJ.

Dans un développement futur du système, nous envisageons de donner à l'utilisateur la possibilité de mettre à jour la table des renversements en temps réel, en jouant de

nouveaux renversements sur un clavier midi connecté à l'ordinateur. Une extension de ce processus de mise à jour pourrait consister à appliquer le schéma d'apprentissage statistique développé par [Assayag 1999]. Ce shéma comporte un algorithme d'analyse incrémental basé sur la technique de compression de Lempel-Ziv. Pendant que le système fonctionne, l'utilisateur joue des renversements correspondant aux accords joués par le système. Puis les caractéristiques stylistiques (textures, enchaînements, etc) de ces renversements sont mémorisés par le système, et ajoutés dans la base de données de renversements pour une sélection future dans le processus d'improvisation.

5. Bibliographies

Assayag G., Dubnov S., Delerue O., Guessing the Composer's Mind: Applying Universal Prediction to Musical Style, *Proc. of the ICMC 99*, Beijing, China (I.C.M.A., San-Francisco, 1999).

Chemillier M., Générateurs musicaux et singularités, *JIM 99, 6èmes journées d'informatique musicale* (CNET-CEMAMu, Issy-les-Moulineaux, 1999) 167-177.

Chemillier M., Grammaires, automates et musique, F. Pachet (éd.), *Informatique musicale* (Hermès, Paris, à paraître 2001).

Coker J., *Improvising Jazz*, 1964, réédition Fireside, 1987.

Dannenberg R.B., Mont-Reynaud B., Following an Improvisation in Real Time, *Proc. of the ICMC 87*, Urbana-Champaign (I.C.M.A., San-Francisco, 1987) 241-248.

Johnson-Laird P., Jazz Improvisation: A Theory at the Computational Level, in: Cross I., Howell P., West R. (eds.), *Representing Musical Structure* (Academic Press, 1991) 291-326.

Pachet F., Surprising harmonies, *JIM 99, 6èmes journées d'informatique musicale* (CNET-CEMAMu, Issy-les-Moulineaux, 1999) 187-206.

Pachet F., Computer Analysis of Jazz Chord Sequences. Is Solar a Blues ?, in: Miranda E. (ed.), *Readings in Music and Artificial Intelligence* (Harwood Academic Publishers, 2000).

Steedman M.J., A Generative Grammar for Jazz Chord Sequences, *Music Perception* **2** (1) (1984) 52-77.

Steedman M.J., The Blues and the Abstract Truth: Music and Mentals Models, in: A. Garnham, J. Oakhill (eds.), *Mentals Models in Cognitive Science* (Erlbaum, Mahwah, NJ, 1996).

Ulrich J.W., The analysis and synthesis of jazz by computer, *Fifth International Joint Conference o Artificial Intelligence* (1977) 865-872.

D'Accord Guitar: an Innovative Guitar Performance System

**Giordano Cabral, Izabel Santana, Rodrigo Lima,
Hugo Santana and Geber Ramalho**

Centro de Informática - Universidade Federal de Pernambuco
Caixa Postal 7851 - CEP 50732-970 - Recife (PE) - Brasil
{grec,izs,rqcl,hps,glr}@cin.ufpe.br

Abstract

Computers multimedia resources may be used to improve musical notation completeness, providing equally an intuitive user interface. An effort in this direction is the development of the so-called instrumental performance systems (IPS), which show the performance directly on a virtual instrument displayed on the computer screen. Unfortunately, the current IPS exhibit various limitations. For instance, they are hardly editable, or not editable at all, they are not enough interactive and their interface is not fully adequate for string instruments. This paper presents D'Accord Guitar, an innovative environment for learning, editing and performing music for guitar. D'Accord Guitar combines various features of the current IPS and other musical tools in order to provide a user-friendly multi-function interface.

Keywords: Musical editing and publishing systems, musical performance modeling and simulation, formalization and representation of musical structures.

1. Introduction

Musical notations try to find the best trade-off between richness and simplicity. The more complete is the notation, the more accurate it is. Rich notations are suitable for situations where the execution precision is essential. This is the case of classical western music, which is based on a score, a notation containing several performance details. On the other hand, the richer is the notation, the less readable it is. Amateur public usually demands simple and intuitive notations for making the learning process faster and easier. Trying to reach this public, notations that represent each music element separately (melody, lyrics, rhythm, harmony) have come out (Wet, Howel & Cross 1991). Using these notations, musicians can focus on each element independently, facilitating learning. In certain music styles, such as jazz, pop, bossa nova, rock'n roll, blues, some of these elements are even omitted, supposing that the musicians already know them. For instance, jazz chord grids represent only harmony and real/fake books represent harmony plus melody (Sher 1991).

Multimedia resources, however, can improve the notation completeness without loosing too much simplicity (Roads 1996). An effort in this direction is the development of the so-called instrumental performance systems (IPS). These systems show directly the performance on a virtual instrument displayed on the screen.

Unfortunately, the current IPS exhibit various limitations. First, they are hardly editable, or not editable at all, not allowing the users to write and play their own or favorite songs. Second, they are not enough interactive, considering the possibilities of teaching musical concepts. Third, the current IPS interface of the string instruments is not fully adequate, since problems such as chord positioning and fingering are not well addressed. Chord Dictionary and Automatic Accompaniment Systems overcome some of these IPS limitations. However, since they have other purposes, they do not fulfill IPS requirements.

This paper presents D'Accord Guitar, an innovative environment for learning, editing and performing music for guitar. D'Accord Guitar combines various features of the current Instrument Performance, Automatic Accompaniment, Chord Dictionary and Karaoke Systems

in order to provide a user-friendly multi-function interface. D'Accord Guitar represents a 2-year designing and programming effort, which required the solution of several interesting problems ranging from chord fingering, to chord positioning, through musical structures representation.

However, the purpose of this paper is to give an overview of D'Accord Guitar. A detailed specification is out of the scope of this paper. Next section presents some current tools and their limitations. Section 3 shows the guitar specific difficulties and constraints. Section 4 presents the principles, general architecture and enumerates the main difficulties in designing and implementing the system, sketching the proposed solutions. The final section draws some conclusions and presents directions for future work.

2. State of art

The most direct way of learning how to play a musical instrument is probably by observing a teacher playing it. Based on this hypothesis, some systems, generically classified as IPS, have been developed. The following sections analyze them and other related systems, explaining their benefits and limitations.

2.1.Instrumental Performance Systems

IPS simulate a human instrumental performance, showing it on a virtual instrument on the computer screen. They also offer some functions for changing the key and the tempo, for searching music on the Internet and for exhibiting extra information about the music (e.g., authors, publisher, etc.). Such systems are built to teach specific songs to people that already have some experience with the instrument, without caring about music theory or instrumental techniques. Examples of IPS are: *iSong*[1], *The Guitarist*[2] and *Desktop Guitarist*[3]. These systems show how multimedia resources can help musical learning process, improving notation accuracy, completeness and simplicity through a direct exhibition of the music performance on virtual instruments. Figure 1 shows the main interface of The Pianist.

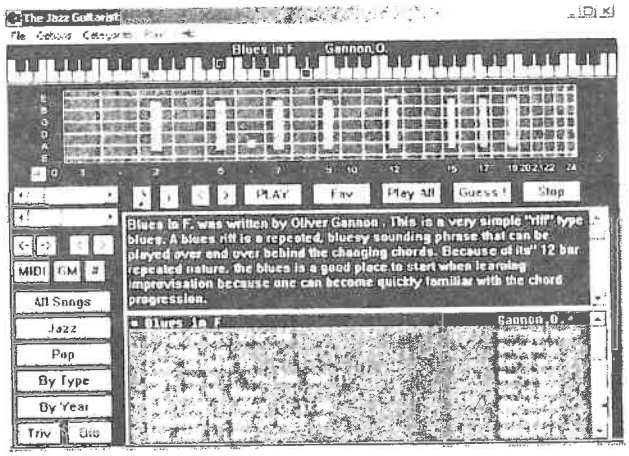


Figure 1 - PG Music The Guitarist

Unfortunately, various criticisms may be addressed to the current IPS. First, the *interactivity* is poor since they do not explore the huge possibility of teaching musical concepts, such as chord construction, recognition, ciphering and fingering (Birmingham & Pardo 2000). The user acts like a spectator, executing only macro commands, such as play,

[1] Inside Music - http://www.isong.com
[2] PG Music - http://www.pgmusic.com
[3] Desktop Music - http://www.desktopmusic.com

stop, rewind, fast forward, change key and change tempo. A higher level of interactivity can be found in chord dictionary systems (like Chord Wizard[4] and Violão Virtual[5]). However, they are not meant to show music performance on a virtual instrument.

Second, the IPS interface for *string instruments,* in particular guitar, does not show the information appropriately, complicating the interpretation of the song being played on fretboard[6]. In fact, the current IPS stress the melodic (solo) information, hiding the underlying harmony (chords). For instance, Figure 2 shows a sequence of notes of an Am7 arpeggio as displayed on a guitar fretboard by a conventional IPS. Since neither the chord cipher is synchronically shown nor the position of all fingers involved in playing sequentially the chord notes are exhibited, it is hard to recognize the harmony. This recognition task is almost impossible when various arpeggios are chained and played fast (Holdsworth 1998).

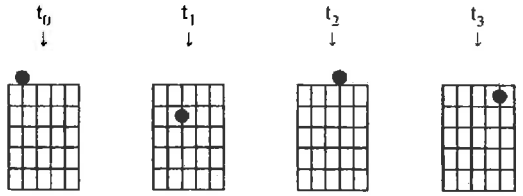


Figure 2 - Arpeggio of a Am7 in a conventional IPS

Third, in all current IPS, the music performance is not *editable*, not allowing users to create or edit their own or favorite songs. The exception is Desktop Music's systems, which are editable but are impractical since the user is obliged to specify every note to be played.

Fourth, several IPS do not include the songs' *lyrics*. Even in the cases where the lyrics are shown, they are not temporally synchronized with the chord ciphers and the melody. As observed in conventional songbooks (Chediak 1999), the fact of displaying lyrics, chord ciphers and melody synchronically, can in principle help users to learn to play a song.

2.2.Automatic Accompaniment Systems

Automatic Accompaniment Systems (AAS), such as Band-In-A-Box[7] and Harmony Assistant[8], are systems that automatically generate melodic lines (e.g., saxophone and bass), rhythmic lines (e.g., drums) and/or chord voicing chaining (e.g., piano) in a particular music style according to a given chord grid (Ramalho, Rolland & Ganascia 1999). Such systems may include an "IPS module", to exhibit the generated music on virtual instruments (usually a keyboard), and partially address some of the discussed problems. For instance, they display the guitar chords ciphers synchronically, facilitating interpretation of notes played on the fretboard. The user can edit song ciphers on a chord grid and associate melody and lyrics to it.

However, since their goal is basically to generate MIDI (Moog 1986) sequences to be used by composers, arrangers and musicians in general, they do not care about performance details. For instance, there is no indication of the possible positions of a chord and its fingering on a guitar neck. This limitation inhibits a suitable use of these systems for string instruments, despite their popularity. Another problem found is the AAS weak flexibility, since, in the majority of these systems, it is impossible to edit the rhythm and/or the chord

[4] Chord Wizard - http://www.chordwizard.com
[5] Violão Virtual - http://www.violaovirtual.com
[6] String instrument neck — in opposition to keyboard
[7] PG Music - http://www.pgmusic.com
[8] Myriad Software - http://www.myriad-online.com

positions. The user must accept the generated accompaniment without being able to perform adjustments.

3. Problems of modeling guitar performance

It is difficult to develop IPS for string instruments, due to its intrinsic ambiguities and spatial constraints. These difficulties are often found in guitar, used as the basis of this study. However, the solutions proposed can be extended for others string instruments, since most of the basic concepts are the same. The first fundamental problem in guitar modeling is that the same pitch may be generated using different combinations of positions[9]. For instance, the 3 positions shown in Figure 3 produces the same pitch.

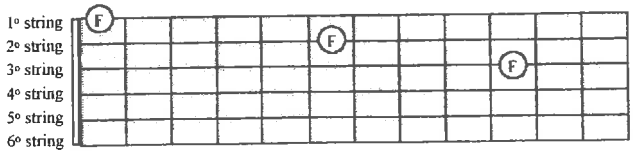


Figure 3 – 3 different positions for the same pitch

For this reason, a transcription from the conventional notations, such as scores, to the actual performance is not straightforward. This transcription problem is even more complicated when converting chord ciphers into actual fretboard positions, since ciphers introduce more ambiguity.

The second fundamental problem is that there are a maximal number of strings to play and fingers to use. These spatial and anatomic constraints help to reduce the number of possible positions of a chord, by forcing the elimination or repetition of notes. However, they are difficult to be modeled.

3.1. Transcribing and playing solos

The conversion from selected positions in the fretboard into notes in a melodic notation is not considered a problem, since each of these positions have a unique correspondent note. However, the task of converting a melodic notation into an actual guitar performance is considered a problem. Fortunately, some algorithms (Sayegh 1989) may be used to satisfactorily solve this problem.

3.2. Recognizing and playing chords

If for single notes there are conversion difficulties, for chords there are much more. Even for a keyboard, there are more than one way of playing a chord: some notes may be doubled, some notes may be omitted, chords have different inversions. Such a problem is called voicing (Fowler 1984a). In the case of the guitar the problem is much harder, since there are also different positions to the same voicing.

In fact, there are two problems. One problem is "how to recognize a chord cipher based on an ensemble of notes selected in the fretboard?". In spite of the fact that there are still open cases (e.g. when to recognize a chord as a Em7(b5) or as a Gm6?), this problem is already well explored, as can be seen on several studies (e.g. Pachet, Ramalho & Carrive 1996).

On the other hand, the task of finding the best guitar performance based on a sequence of chord ciphers is much harder (Fowler 1984b). This is difficult because there is not a single

[9] In this paper, the word *position* indicates a fretboard position (a tuple <string, fret>). A *chord position*, however, indicates a set of positions that composes a chord.

possible best conversion, since there are multiple parameters that can be taken to choose the best chord positions. To a better understanding of this problem, it will be divided in subtasks. It is necessary to know all possible positions for a given chord and to evaluate them. To explore all possible chord positions for a given chord it is necessary to find out not only the frets and strings used, but also the fingers used. So, there are two distinct problems. The first is called positioning, and consists in finding which are the frets and strings used in each chord position. The second is called fingering and consists in find out what finger is used in each pressed string (Cordier 1995). In the following sections these problems will be described more deeply.

Positioning

The complexity involved in the positioning problem can be observed analyzing it combinatorially. Assuming a system that allows positions from fret 0 to fret 12, each string can be used to play any note of a chord (since each string covers a whole octave). Given a chord composite of n notes, there are at least $6n$ fretboard positions that may be used to form such a chord position. In Figure 4 can be seen the possible positions for a Fm7(11).

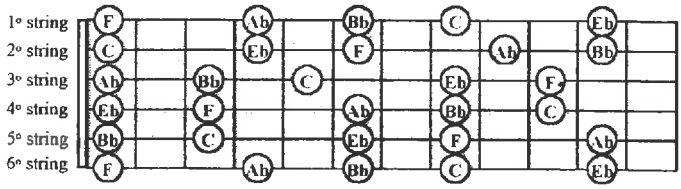


Figure 4- Possible positions for a Fm7(11)

Using all 6 strings, there are n^6 combinations of such notes (that may eventually compose the chord). Using only 5 strings there are $6n^5$ combinations of positions, and so on. This shows the large amount of data to deal with. For the Fm7(11) instance, there are a total of 43750 combinations of positions that may compose a chord. Such combinations must be filtered according to voicing and anatomic constraints.

The voicing constraint filters which of these combinations really compose the desired chord. In the previous case, only 5880 of the 43750 are in fact considered Fm7(11). Figure 5 shows some of these Fm7(11) positions, each one with a different voicing.

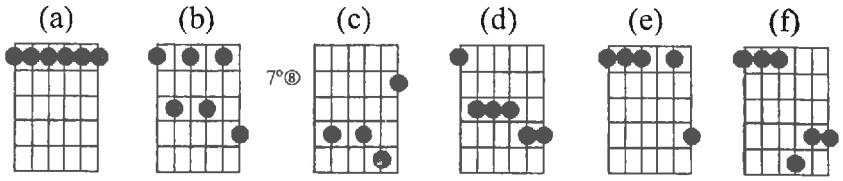


Figure 5 - Some positions of Fm7(11) in a guitar fretboard. 6^{th} string on the left, 1^{st} string on the right. Circles indicate the positions (fret 0 up). Strings without circles are unavailable.

Fingering

The complexity involved in the fingering problem is due to a difficult computational modeling of the anatomy of human hands. This modeling is necessary to find which fingers to use in each pressed string, and, consequently, to filter positions impossible to perform. This filtering is based on left-hand and right-hand constraints. The right-hand filter assumes that the player always plays his/her 3 fingers in consecutive strings, avoiding positions such as (e) of Figure 5. The left-hand filter avoids chord positions where there is a big distance between fingers (such as (f) of Figure 5) or that needs more than 4 fingers to be played (such as (d) of

Figure 5). In the Fm7(11) case, the right-hand filter eliminates 1296 of the 5880 chord positions. After the left-hand filter execution, only 283 chord positions remains.

After finding the positions, it is necessary to search the possible left-hand fingers to put at each one, including the possibility of using a bar chord. The amount of possibilities involved in fingering is extremely smaller, but the modeling is even harder. To demonstrate it, lets use the basic position of Dm7, shown in Figure 6. In such a case, there are 24 combinations of fingers that do not use a bar chord, plus 3 combinations of fingers that uses it. Figure 6 shows some of them.

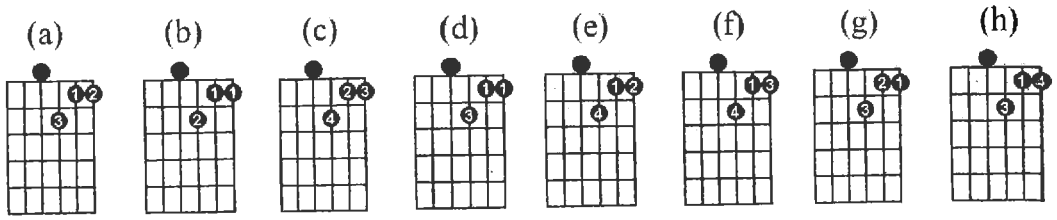


Figure 6 – Some possible fingerings of the Dm7 basic position. Numbers indicates the fingers used. Circles without number indicates a not pressed string. Strings without circles are unavailable.

This figure shows the difficulty of automatically finding values to its attributes. The Figure 6a and Figure 6b fingerings are the most commonly used (the first one does not use a bar chord while the second one does). The Figure 6c, Figure 6d and Figure 6e fingerings are less used, but their use is justifiable depending on next chord positions and guitar player styles. For instance, if next chord is the Em7 shown in Figure 7, Figure 6c fingering is justifiable.

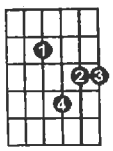


Figure 7 – Fingering for a Em7 position

Additionally, Figure 6f fingering is unjustifiable, Figure 6g fingering is unused and Figure 6h fingering is impossible to perform. Concluding, for the 27 possible fingerings of the Dm7 position shown in Figure 6, 7 are executable, but only 2 are used in most cases. To an IPS, the problem is how to find these 7 executable fingerings and how to evaluate them to choose the best one depending on the context.

Best positioning and fingering for chord chaining

After generating all possible chord positions and their possible fingerings, it is necessary to find and weight up parameters to evaluate them. Some of these parameters are context independent, such as comfort, commonness and flexibility, but it is hard to measure them in a deterministic way.

Nevertheless, this context free evaluation is not reasonable. The final objective is to play a chord grid. Then, the system must take into account all chords involved. Then, it is necessary to consider context dependents parameters, such as proximity of the bass note, similarity of chord positions, etc. The searching of these best positions and fingerings based on a chord grid is the hardest problem, and is called in this paper "*best positioning and fingering in chord chaining*".

To our knowledge, there is no available solution for this problem. The main innovation of D'Accord Guitar is to have developed a solution for it.

4. D'Accord Guitar

This section discuss the main design choices made in developing D'Accord Guitar, as well as the proposed solutions for some of the previously discussed problems. However, a detailed explanation about these solutions is not in the scope of this paper.

4.1.Principles

The basic assumption in D'Accord Guitar is that a proper separation of melody, harmony, rhythm and lyrics information improves user learning, letting him/her focuses on a particular element at once. For example, a chord accompaniment can be seen not just as an ensemble of notes sequentially played, but as a rhythm being applied to a sequence of chords, generating such notes. Instead of learning a sequence of notes, the user could then focus only on a chord sequence associated with a known rhythmic pattern.

This separation is also of great help in the song edition process, since it will be possible to edit each element separately, omit some elements and generate some elements automatically. For instance, the user can specify the chord grid, import the melody from a MIDI file, and specify a rhythmic pattern to be applied on the chords. In particular, the explicit representation of the underlying harmony can solve the problem illustrated in Figure 2. In fact, knowing the chord, it is simple to show the position of all fingers involved in playing it. This separation can be a basis for an adequate approach of problems such as chord positioning, fingering and chaining. For instance, when playing an arpeggio of a chord, D'Accord Guitar does not show an animation note by note, as a traditional IPS do. The position of every finger is indicated in the fretboard during the chord time span. At the moment that a note is played, the correspondent fingers (of the left and right hands) are highlighted, as illustrated in Figure 8.

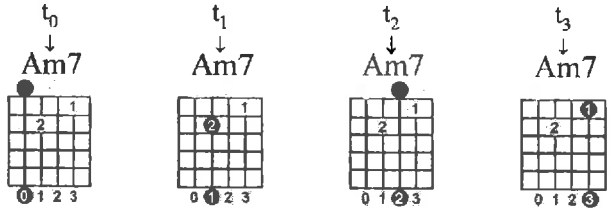


Figure 8 - Arpeggio of an Am7 in D'Accord Guitar. Circles at the top indicate left-hand fingers. Circles at the bottom indicate right-hand fingers. Black circles indicate the played string. Gray circles indicate finger that are not playing, but that are positioned. Strings without circles are unavailable.

4.2. User interface

Regarding the problems of interactivity, information displaying, editing capabilities, and flexibility, discussed in Sections 2.1 D'Accord Guitar proposes some solutions and/or improvements.

As all the IPS, the didactic concern of D'Accord Guitar is reflected by some controls, like loop and tempo change, and by a visible fretboard. However, the interactivity provided by our system goes beyond these simple features. In D'Accord Guitar, there are three different user interaction modes, for executing a song, editing a song and browsing chords/rhythms.

In the *executing mode*, the user can perform executing (playing) commands over the song (such as play, stop, pause, loop, fast forward, rewind, change tempo and change song position). Figure 9 shows the main interface of D'Accord Guitar's in such a mode. In the central part of the window, there is the virtual guitar neck indicating the left and right fingers positions from the guitar player standpoint (guitar's nut on the left, 1st string up, 6th string down). When a string is played, the correspondent fingers pictures are highlighted (in Figure 9, fingers 1, 2 and 3 are playing simultaneously). The bottom of the window exhibits song's ciphered chords and lyrics, animated as in Karaoke systems, such as WinOKE[10] and RealOrche[11]. D'Accord Guitar has generic functions for file manipulation and MIDI configuration. Users can also transpose the song, see other positions of a given chord and toggle between the harmony exhibition mode (where chords and all fingers involved in playing them are shown) and the solo one (where the song melody is played on the fretboard as a solo).

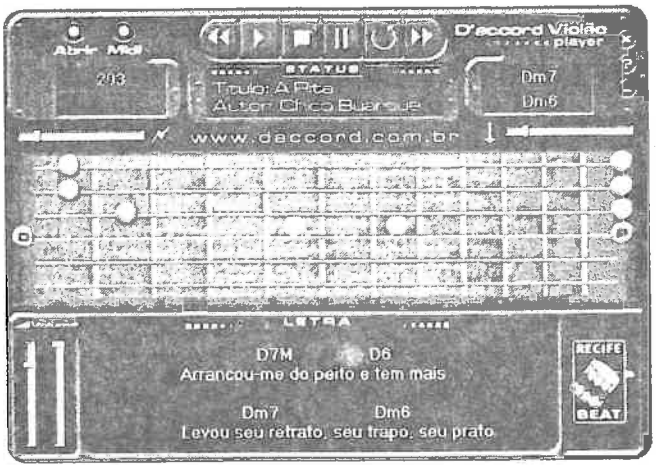


Figure 9 - D'Accord Guitar Interface in execution mode

In the *edition mode*, the user can create a song from scratch or edit an existent one, writing the chord grid, lyrics and recording each song element. Each element of the music is recorded or edited separately. The melody can be recorded via a MIDI instrument or obtained via a MIDI file. The harmony is obtained from a chord grid (like in Band-In-A-Box), with the flexibility of changing the chord position according to the user demand (e.g., in the case of Figure 9, the user may prefer a Dm7 in the fifth fret). The rhythm can be recorded from a MIDI instrument or from the computer keyboard (see Figure 10), or it can be chosen from the rhythm base. Since, each element is recorded separately, the system implements studio recording mechanisms of overdubbing, punch in and punch off (Keating & Anderton, 1998), so that a new element can be recorded in real time while the other already stored elements are played synchronically.

Moreover, it is possible to create and store new chords and rhythmic patterns, and to assign a specific fingering and position to a chord. Finally, during edition, the user may apply a stored rhythmic pattern to a group of chords or, alternatively, directly record the rhythm of the whole song while he/she is playing on the MIDI keyboard, MIDI guitar or computer keyboard (see Figure 10).

[10] http://www.winoke.com
[11] http://www.realorche.com

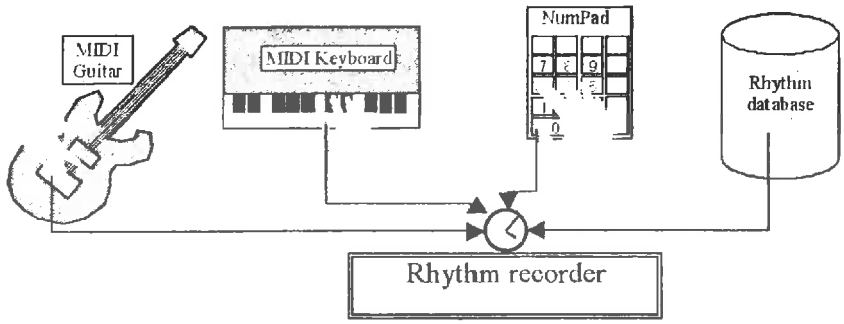


Figure 10 - General architecture of the rhythm recording module

In the *browsing mode*, the user can interact directly with the fretboard and browse the chord and rhythm databases. The user can stop the system performance in order to browse musical concepts (typically chords) by interacting directly with the fretboard. For example, while the user chooses chord notes in the fretboard, the program is able to show the notes names, the chord cipher, the intervals between the notes and chord the root, and the fingers that must be used to play these notes. The user can also navigate through chord and rhythm databases, as in a chord dictionary system.

4.3.Architecture

Figure 11 shows D'Accord Guitar general architecture. Thick lines indicate the execution mode interactions, dotted lines indicate the edition mode interactions and the thin lines indicate the browser mode interactions.

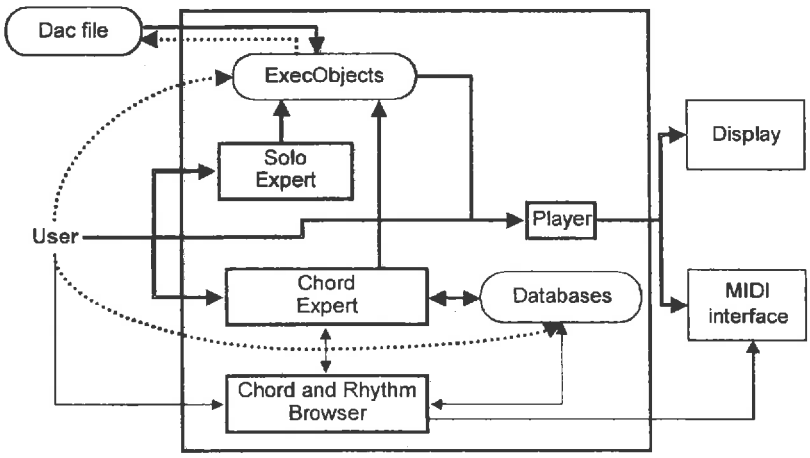


Figure 11 - D'Accord Architecture

The ExecObjects in Figure 11 are the executable musical objects. They are modeled as events, synchronized in order to play the song. There are harmonic, melodic, rhythmic, lyrics, solo and control events, each one with its own attributes. D'Accord Guitar has a proprietary file format to store or load these events, shown in Figure 11 as the Dac File.

The Browser module allows the user to browse the chord and rhythm databases and to interact directly with the fretboard.

The Chord and Solo Expert modules are designed to solve problems such as right and left hand fingering, chord voicing and positioning. The Solo Expert deals with transcribing and playing solos, and uses existent algorithms (Sayegh, 1989; Cordier, 1995). In contrast, the Chord Expert presents innovative solutions, which will be discussed in next section.

4.4.The Chord Expert

The Chord Expert is responsible for recognizing and playing chords. As explained previously, the main problem is the *best positioning and fingering in chord chaining*. In this section will be discussed a propose to solve this problem, partially implemented in D´accord Guitar.

Positioning and Fingering

Before finding the best chord position, Chord Expert needs to know all possible positions of each chord (enabling positions ranging from fret 0 to fret 12). D´accord Guitar performs an exhaustive search (Russel & Norvig 1994), filtering the results according to voicing and fingering constraints. The voicing constraint ensures that the chord position actually composes the chord (i.e. only the 5th may be omitted). The fingering constraints ensure that the chord position is executable. For instance, the chord position must not need more than 4 fingers to be placed and the maximum distance between fingers in the chord position must not exceed 4 frets.

For each chord position generated, D´accord Guitar calculates the possible left-hand fingerings, including the possibility of using a bar chord. This is a constraint satisfaction problem, and its solution is based on simple heuristics. For instance, if there are two strings pressed on different frets, the string using the lowest fret uses the lowest finger. If there are two strings pressed on the same fret, the upper string uses the lower finger. In Figure 12 the finger 1 is placed on the 2nd string and the finger 2 is placed on the 4th string by the first rule. The finger 3 is placed on the 5th string and the finger 4 is placed on the 3rd string by the second rule.

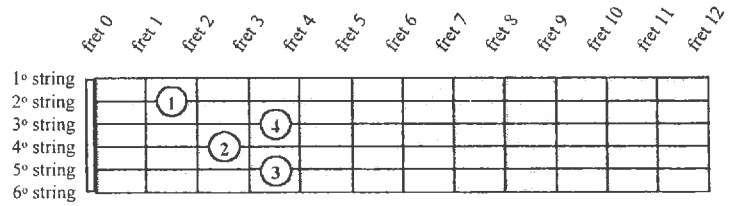


Figure 12 – a possible Db7 position in the fretboard

After the search, the system suggests values for the evaluation parameters like comfort and flexibility. These parameters are calculated based on the distance between fingers, the number of finger used and the hand orientation. These parameters together with the correspondent voicing, produces a ranking of the result.

The search done generates an amount of data impossible to deal with in a real-time application. Moreover, this data does not change after generated. For these reasons, this algorithm is executed offline and its results were stored in the Chord Database.

Best positioning and fingering in chord chaining

To find the best positions and fingerings taking into account the other chords, the system uses a hybrid model, joining a heuristic search (Rayward, Osman & Reeves 1996) with Case-Based Reasoning (Kolodner 1993). The search is based on a multi-attribute (Russel & Norvig 1994) function using different and often-contradictory parameters that can be weight up by the user. There are context-free and context-dependent parameters. Context-free parameters refer to questions like: "*how usual is the chord position/fingering?*" and "*how easy is to play the chord position/fingering?*". Context-dependent parameters refer to questions like: "*what are the chord positions where there is the smoothest bass line?*". The

Case-Based Reasoning measures the similarity between chord positions, and is also useful to transpose a song, trying to keep the chord positions as similar as possible with the original ones.

This feature is still being validated. To illustrate the current state of it, we performed a simple test. Given the chord sequence: Em / C7M / D7 / G, the results when usual chord are preferred (**Erreur ! Source du renvoi introuvable.**) are different from those where a "smooth" bass line (possibly with unusual chord positions) is preferred (Figure 14).

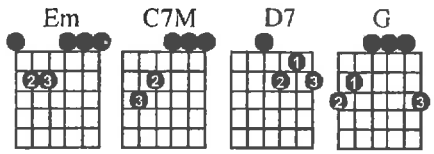


Figure 13– Most usual positions and fingering for the given chord grid

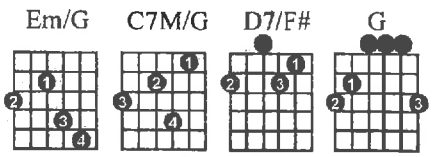


Figure 14 – Chord positions with the closest bass notes

These results are not conclusive yet, since we are assessing just one of the evaluation parameters at each time. However, the first results are encouraging, indicating that this approach can lead us to a complete and flexible solution to the problem presented.

5. Conclusion

This paper presented D'Accord Guitar, a learning, editing and performance tool, specific for guitar. D'Accord Guitar can be seen as an Instrumental Performance System with an original underlying representation of guitar musical structures. Based on this representation, some editing functions existing in Automatic Accompaniment Systems have been adapted and integrated. Beyond the proposed representation and the editing capabilities, D'Accord Guitar environment provides special mechanisms for helping amateurs to learn musical concepts. These learning mechanisms and the advanced editing functions qualify this system to be used by both beginners and skilled guitar players.

Our future works include the improvement and validation of the solutions proposed to some problems, such as transposition and fingering. Although most of the system has already been implemented and tested, it has not been concluded yet, in particular the editor modules. We intend to finish the implementation and perform a broad validation. Finally, we plan to adapt this system to other different string instruments, since their underlying representation structures are probably similar.

6. References

Birmingham, W. & Pardo, B. (2000) On The Computational Properties of Harmonic Analysis. In the *Proceedings of the Workshop on Artificial Intelligence and Music*. AAAI'2000.

Chediak, A. (1999). *Songbook of Chico Buarque* (vol 1-4). Rio de Janeiro: Lumiar Ed.

Cordier, M.-O. (October, 1995) Doigtage intelligent d'une partition de guitare. N° 23, 46-48. Bulletin de L´AFIA.

Fowler, W. (1984a) *Chord Voicing Systems*. Fowler Music Enterprises.

Fowler, W. (1984b) *Chord Progression Systems*. Fowler Music Enterprises.

Holdsworth, A. (1998) *Melody Chords for Guitar*. New York: Hal Leonard.

Keating, C. & Anderton, C. (1998) *Digital Home Recording - Tips, Techniques, and Tools for Home Studio Production.* Backbeat Books.

Kolodner, J. (1993) *Case-Based Reasoning.* San Mateo: Morgan Kaufmann.

Moog, B. (1986) MIDI: Musical Instrument Digital Interface. *Journal of the Audio Engineering Society*, 34(5), 394-404.

Pachet, F., Ramalho, G., & Carrive, J. (1996). Representing Temporal Musical Objects and Reasoning in the MusES System. *Journal of New Music Research,* 25(3), 253-73. Swets & Zeitlinger.

Ramalho, L., Rolland, P-Y., Ganascia, J-G. (1999). An Artificially Intelligent Jazz Performer. *Journal of New Music Research, 28(2), 105-129. Swets & Zeitlinger.*

Rayward-Smith, V., Osman, L & Reeves, R. (1996) *Modern Heuristic Search Methods.* John Wiley & Son.

Roads, C. (1996) *The Computer Music Tutorial.* Massachusetts: MIT Press.

Russel, S & Norvig, P. (1994) *Artificial Intelligence: A Modern Approach.* Englewood Cliffs: Prentice Hall.

Sayegh, S. (1989) Fingering for String Instruments with the Optimum Path Paradigm. *Computer Music Journal*, 13(3).

Sher, C. (1991) *The New Real Book* (vol. 1 and 2). Berkeley: Sher Music.

Todd, P & Loy, G. (1991) *Music and Connectionism.* Massachusetts: MIT Press.

West, R., Howell, P., & Cross, I. (1991). Musical Structure and Knowledge Representation. In P. Howell, R. West, & I. Cross (Eds.), *Representing Musical Structure* (pp. 1-30). London: Academic Press.

Guitare électrique, nouvelle génération

François Pachet
SONY CSL-Paris, pachet@csl.sony.fr

Abstract : Nous décrivons un système d'aide à l'improvisation musicale qui permet d'étendre les capacités techniques d'un musicien en produisant des phrases de continuation. Ces continuations sont conformes au style du musicien, style lui-même appris par le système de manière agnostique, en continu. Le système s'oppose aux systèmes de génération musicale automatiques sur plusieurs aspects techniques (suivi de l'harmonie). Il ouvre la voie a plusieurs modes de jeu musicaux collectifs nouveaux, par exemple en permettant aux musiciens d'échanger leurs base stylistiques en temps réel.

1. Introduction

L'improvisation musicale, le Jazz en particulier, est une activité aussi fascinante que frustrante. L'acte d'improvisation met en œuvre, voire à l'épreuve, la relation intime entre pensée musicale et mouvement : l'improvisateur doit tout à la fois écouter, penser, trouver des idées musicales et *in fine* effectuer les gestes musicaux appropriés. La vitesse de ce processus, et le manque de temps pour réaliser cette chaîne de la pensée au geste est dans ce contexte un ingrédient central, c'est probablement ce qui rend l'improvisation excitante. C'est aussi ce qui peut la rendre frustrante, puisque les musiciens, débutants ou professionnels, sont, par définition, limités dans leur habileté technique ainsi que par la morphologie de l'instrument : ça ne va pas toujours aussi vite que l'on voudrait. Cette frustration peut être négative ou positive, par exemple source de raccourcis ou de nouveauté. C'est en tout cas au musicien de décider comment utiliser sa technique ou son manque de technique de manière appropriée.
Nous proposons de concevoir des instruments de musique qui permettent au musicien - d'une certaine manière - d'étendre ses possibilités techniques tout en respectant les règles du jeu de l'improvisation : temps réel, rapidité et surtout primauté du contrôle au musicien, qui doit, qui veut, en dernier ressort, décider et « réaliser » sa pensée.

De nombreux chercheurs en informatique musicale ont abordé le « problème » de l'improvisation. D'un point de vue technique il s'agit la plupart du temps de fabriquer des improvisateurs automatiques en utilisant toute la panoplie des techniques informatiques disponibles (Biles, 1998 ; Ramalho, 1994 pour n'en citer que deux). Certains travaux ont par ailleurs l'ambition de comprendre le processus « cognitif » de l'improvisation, via la conception de modèles et leur expérimentation (Johnson-Laird, 1991). Malgré quelques succès dans des cas très particuliers (le bassiste de Jazz de Ramalho par exemple, cf. Pachet, 2000) les performances de ces systèmes restent très en deçà des performances humaines, et le sentiment d'écouter un automate, au sens primitif du métier à tisser de Vaucanson, est indélébile. En outre, et comme les cartes perforées fournies à ce même métier à tisser, ces systèmes nécessitent, pour fonctionner correctement, l'apport de connaissances humaines relativement complexes et le plus souvent entrées à la main (validation pour les systèmes d'apprentissage supervisés, séquence d'accords sous jacente, tempo, structure du morceau, etc.), ce qui compromet encore leur usage dans des situations de jeu réelles.

Le système que nous présentons ici se distingue de ces approches par plusieurs aspects fondamentaux. D'une part notre système n'est pas autonome, mais au contraire s'intègre de

manière intime au jeu d'un vrai musicien, par opposition aux automates décrits plus haut ou aux systèmes de question/réponse (Biles, 1998 ; Wessel, 1991). Par ailleurs, nous utilisons un module d'apprentissage non supervisé qui ne nécessite pas de connaissances humaines explicites, ni de connaissances sur le morceau (harmonie, tempo, structure, etc.), et s'adapte automatiquement aux changements imprévus - de style, tempo, etc. Le système résultant doit être considéré et le cas échéant évalué, non pas comme un système autonome, mais comme un couple musicien / ordinateur. Une de ses caractéristiques principales est en effet, comme on le verra plus bas de substituer les connaissances symboliques explicites des systèmes traditionnels par du contrôle.

2. Description du système

Notre système est constitué de deux modules, fonctionnant tous deux en temps réel: un module d'analyse et un générateur de phrases. Ces modules prennent en entrée des flux d'informations Midi. En principe, toute information Midi peut être traitée par ces modules, mais nous n'avons utilisé pour l'instant que les hauteurs (pitch) et vélocité. Les expérimentations réalisées ont utilisé une guitare Midi - d'où le titre - dans un contexte Jazz, mais sont aisément transposables à d'autres instruments Midi (clavier, saxophone, violon) et d'autres styles.

Continuation et non pas réponse

Plusieurs modes de jeu peuvent être envisagés avec le système : le nombre de musiciens peut être arbitraire, et pour chaque musicien, il existe plusieurs modes de contrôle individuel (voir 4.2). Dans le mode par défaut, à un seul musicien, le jeu en temps réel du musicien est donné en entrée du système, et la sortie du système est envoyée à un synthétiseur dont la sortie est mixée avec l'entrée. Dans nos expérimentations, nous avons utilisé le même son pour le jeu « humain » et la sortie de système, rendant ainsi la contribution du système pratiquement indistinguable de celle du musicien (voir plus loin à ce sujet).

Dans ce mode standard, le système joue le rôle d'un continuateur de séquence, à la manière des mécanismes de *sequence completion* sur certains systèmes d'exploitation. Le jeu du musicien humain est continuellement segmenté en phrases musicales, en utilisant un seuil temporel (de l'ordre de 200 millisecondes). Dès qu'une phrase est détectée (i.e. un silence plus long que le seuil), celle-ci est envoyée en même temps à l'analyseur et au générateur. L'analyseur construit ainsi incrémentalement un modèle des phrases récurrentes du musicien. Le module de génération tente, lui, de compléter la séquence d'entrée en utilisant le modèle construit par l'analyseur. Ce dernier point est important : la séquence générée par le système n'est pas une « réponse », comme dans le système de Biles (1998) par exemple, mais bien une complétion. Musicalement, la différence est en effet de taille puisque dans le cas d'une réponse, on peut considérer que les deux phrases (la question et la réponse) sont bien distinctes (et d'ailleurs souvent jouées, en Jazz, par des instruments différents). Dans le cas de la complétion, il s'agit de la *même phrase*, qui est simplement initiée par le musicien et continuée par le système.

Détection de patterns et l'algorithme LZ

Le module d'analyse (ou d'apprentissage) tente systématiquement de repérer dans les phrases qui lui sont données en entrées des patterns récurrents. De nombreuses techniques existent pour repérer de telles régularités (voir par exemple P. Y Rolland (1999) pour les techniques basées sur la programmation dynamique). Nous avons utilisé pour nos expérimentations une

méthode particulièrement simple, issue du domaine de la compression de données : l'algorithme Lempel-Ziv (Ziv & Lempel, 1978), dont il a été montré qu'il se prêtait bien à la modélisation de patterns mélodiques (Assayag et al., 1999), tout en ayant de bonnes performances. Cette technique consiste à construire un arbre de suffixes (nommé arbre LZ) par une simple lecture de gauche à droite de la séquence d'entrée. Chaque fois qu'un nouveau suffixe est détecté, celui-ci est ajouté à l'arbre, comme fils du sous-pattern dont il est l'extension directe. Le procédé est dans son principe très simple, et est décrit en détail par exemple dans (Assayag et al. 99). Nous l'avons modifié pour l'adapter à notre contexte en deux points importants.

D'une part, nous avons besoin d'une représentation efficace de la technique d'analyse, pour pouvoir être utilisée en temps réel, à la fois pour la mise à jour de l'arbre LZ (analyse) et pour la génération. Assayag et al. ont proposé une représentation duale de l'arbre LZ, qui en théorie permet d'accélérer la procédure de génération, au prix d'un encombrement mémoire très important. Après expérimentations, nous ne sommes pas persuadés que cette technique soit efficace pour les corpus considérés (de l'ordre de 40.000 nœuds). Nous avons opté pour une technique consistant à exploiter l'arbre LZ « standard », augmenté par une table de Hash, permettant d'obtenir directement la liste des nœuds LZ pour chaque donnée musicale (dans notre cas un pitch Midi). Cette table est mise à jour à chaque création de nœud LZ. En phase de génération, pour une séquence commençant par un item i, la table donne ainsi directement la liste des nœuds possibles.

La deuxième modification concerne la nature intrinsèquement incomplète de l'algorithme standard de génération à partir d'un arbre LZ. Pour chaque génération, le système calcule d'abord toutes les continuations possibles, puis en tire une au hasard, de manière pondérée, en fonction des probabilités d'occurrence de chaque continuation. Par construction cependant, l'arbre LZ n'est pas complet (par opposition aux modèles basés sur les chaînes de Markov). Ainsi, un pattern musical donné peut se trouver à plusieurs endroits de l'arbre. Par exemple, le pattern ABC se trouve à deux endroits de l'arbre de la figure 1.

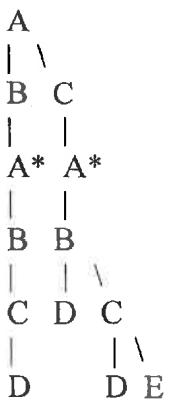


Figure 1 . Un arbre LZ contenant deux fois la sous séquence ABC (chaque début de pattern est indiqué par une étoile).

Pour détecter et traiter ces multiples occurrences, nous avons modifié légèrement l'algorithme de génération standard LZ, en calculant, pour chaque sous séquence d'entrée à « continuer », toutes les continuations possibles, à n'importe quelle position dans l'arbre (au lieu de ne considérer, à chaque fois, que la racine).

Dans notre exemple, pour une séquence d'entrée ABC, la table de Hash nous donne directement les trois nœuds contenant la donnée « A ». Trois traversées de l'arbre sont donc effectuées à partir de chacun de ces nœuds, pour vérifier qu'ils correspondent bien à la séquence ABC. Deux seulement de ces nœuds satisfont cette propriété, et les continuations possibles sont donc {D, D, E} avec comme poids respectifs {1, 1, 1}. Un tirage aléatoire est alors effectué pour choisir la continuation.
Nous obtenons ainsi un système capable de traiter environ 30.000 nœuds en temps réel, c'est à dire avec un temps de réponse pour la génération de moins de 200 millisecondes, avec une implémentation Java utilisant MidiShare (Orlarey et al., 1998) et tournant sur un PC Pentium III 600 Mhz. Le capteur Midi utilisé est un système Roland GK-II / GR30.

3. Remplacer la connaissance par du contrôle

Notre système est limité intentionnellement dans ses connaissances musicales. Les limitations sont de trois types : rythmique, harmonique et polyphonique. Nous pensons que ces limites sont en fait la source même de l'efficacité du système et rendent possible son usage en situation réelle. Nous discutons ici ces trois points.
Premièrement, le système n'a pas de connaissance rythmique, ni pour l'analyse ni pour la génération. Les phrases générées sont toutes des séquences linéaires de double croches (du moins on peut les interpréter ainsi). Quelques expérimentations avec des structures non linéaires ne nous ont pas convaincu que l'on pouvait faire beaucoup mieux dans notre contexte. En effet, la génération par le système de séquences non linéaire est difficile à contrôler, et engendre immanquablement des phrases maladroites, dans le style des automates musicaux. Ce défaut provient probablement du fait que nous ne savons pas bien modéliser le rythme musical autrement que par sa réduction à une partition. En revanche, le système mémorise et génère des informations de vélocité, et les phrases engendrées restituent une impression de naturel indéniable.

Deuxièmement, le système est par essence monophonique, et ne génère donc pas d'accords. Même si techniquement l'arbre LZ peut aisément accommoder des structures harmoniques, quelques expérimentations nous ont convaincu que la polyphonie ne pouvait se traiter comme la mélodie du point de vue du contrôle. La gestion analyse / génération d'accords et de séquences d'accords est traitée par ailleurs (Chemillier, 2001 ou Pachet, 1999) par d'autres techniques, avec un point de vue générationnel tout à fait différent : il s'agit plus de produire des variations sur une même structure algébrique, que de continuer, à proprement parler, des séquences.

Enfin, le système n'a pas de connaissance harmonique. L'harmonie est un point central dans l'improvisation de Jazz car la séquence d'accords sous-jacente à une mélodie permet en quelque sorte de décider quelles sont les notes utilisable à tout instant (avec bien sûr la flexibilité que l'on sait). Notons ici que la détection de l'harmonie sous jacente est relativement aisée à effectuer pour un musicien : celui-ci sait en général assez bien ou il se trouve dans la séquence d'accords ou la mélodie. En revanche, cette information est assez difficile à donner à l'ordinateur. D'une part il faut en effet donner à l'avance la séquence d'accords (ce qui empêche les changements et variations imprévus). D'autre part, il faut indiquer à l'ordinateur où en sont les musiciens, et donc résoudre les nombreux et délicats problèmes de synchronisation en temps réel, suivi de partition et autres.
Même si ces problèmes sont solubles en théorie, nous avons choisi de ne pas représenter d'information harmonique de manière explicite pour trois raisons principales. D'une part il est relativement facile de « corriger » ou plutôt relancer le système lorsque celui-ci quitte trop

brutalement l'harmonie sous- jacente : il suffit de jouer quelques notes (tierce ou quinte) pour relancer le système dans la « bonne » direction. Deuxièmement, le système apprend continuellement toutes les phrases jouées par le musicien. Le plus souvent, les improvisations de Jazz consistent à exploiter une structure harmonique de manière répétitive. Ainsi, les enchaînements harmoniques spécifiques de la grille vont être progressivement « appris » par le système. Un exemple typique de ce phénomène est le morceau « So What » de Mils Davis, qui ne comprend que deux accords (D min et D# min). A force d'analyser des phrases jouées sur cet enchaînement, le système finit par connaître des patterns de transition D min / D# min et donc les rejoue lui-même.
Enfin, nous avons développé un mode de contrôle spécifique qui permet d'influencer la génération par des informations harmoniques extérieures sans analyse harmonique complexe (*lightweight*) et sans complexifier le système. L'idée est d'introduire un biais pour le calcul des continuations, sous la forme de contraintes. Des informations provenant de l'extérieur (par exemple un deuxième musicien jouant du piano Midi) peuvent être envoyées en temps réel au module de génération : typiquement, les 8 dernières notes (pitch) jouées par ce musicien. Ces informations peuvent alors être utilisées pour influencer le choix des continuations, à partir de l'arbre LZ. Au moment du tirage aléatoire, la procédure standard (décrite plus haut) consiste à associer des poids en fonction des probabilités d'occurrence de chacune des continuations possibles. Nous pouvons changer ce schéma, et remplacer ces probabilités d'occurrence par une fonction quelconque des continuations, par exemple en privilégiant les notes qui sont égales ou proches harmoniquement des notes reçues par l'entrée extérieure. L'effet immédiat de ce changement de fonction aléatoire est d'influencer la génération de l'arbre vers la direction harmonique correspondant aux notes extérieures. Ainsi, on peut se rapprocher plus ou moins d'un musicien, tout en gardant le même schéma de contrôle sur le système. Ceci permet alors, in fine, de donner une sensibilité harmonique au système, sans avoir entré d'information symbolique (la séquence d'accords), et avec toute la flexibilité souhaitée (changements de tempo et d'harmonie par exemple).

4. De nouveaux modes de jeu

Le système que nous proposons ici introduit plusieurs modes de jeu nouveau avec un ordinateur.

1. Contrôleurs de base

D'une part, pour faciliter l'usage du système, nous avons développé une série de contrôleurs déclenchables en temps réel pendant le jeu de l'improvisation. Ces contrôleurs permettent de modifier certains paramètres du système, pour les deux modules d'analyse et de génération.
Ces paramètres sont en particulier :

- déclenchement systématique pour chaque phrase détectée (on/off)
- apprentissage systématique des phrases détectées (on/off)
- superposition (on/off). Ce paramètre permet de décider si l'on autorise ou non les superpositions de phrases musicales. Par défaut, le système s'interrompt dès que le musicien commence une nouvelle phrase. Avec un peu d'entraînement, on peut ainsi arriver à jouer de longues phrases « sans couture » mélangeant les contributions du musicien et de l'ordinateur.

Tous ces contrôles sont déclenchables à partir d'un pédalier Midi. Des expérimentations avec des gants Midi sont en cours.

Par ailleurs, un certain nombre de paramètres hors temps réel permettent d'adapter le système à un style donné et sont accessibles via une interface graphique simple (Cf. Figure 2) : le nombre de notes généré par le système (multiple du nombre de notes de la phrase d'entrée), le tempo (idem), le nombre de fois que chaque séquence d'entrée est apprise (ici, 3), et l'apprentissage éventuel de transpositions de la séquence.

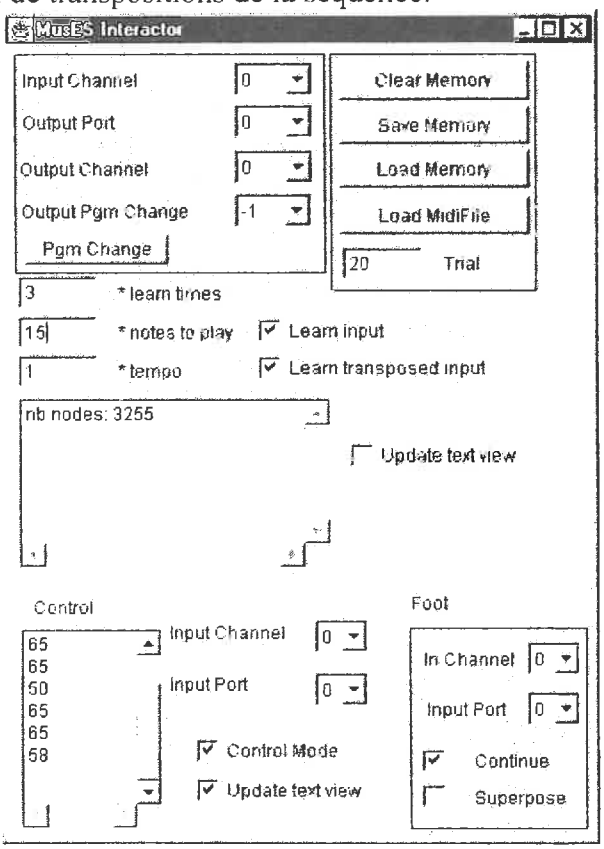


Figure 2. L'interface graphique du système et les paramètres non temps réel.

2. *Modes de jeu*

Une conséquence directe de la conception du système, séparant module d'analyse et de génération, et qu'il permet de considérer de nouveaux modes de jeu pour l'improvisation. Nous listons ici quelques-uns de ces modes :

- *Autarcie simple*. C'est le mode par défaut, décrit ici : un seul musicien avec un seul système. L'apprentissage est effectué off-line, à partir du jeu du musicien (sauvegardable par fichier) ou d'un ensemble de MidiFiles. Nous avons en particulier expérimenté avec des MidiFiles de chorus de Pat Martino (Hauser, 1994).
- *Autarcie multiple*. Dans ce mode, chaque musicien a son propre système. C'est la simple démultiplication du précédent, mais chaque musicien reste autarcique.
- *Maître / Esclave*. Dans ce mode, un musicien utilise le système par défaut, un autre produit une influence extérieure, par exemple harmonique, comme décrit dans la section 3. Ce schéma de base est lui-même cumulable avec d'autres, voire peut être mis en boucle.
- *Cumulatif*. Ici plusieurs musiciens utilisent le système en partageant simplement la base de patterns. Ceci permet alors de constituer une base commune musicale, et de jouer collectivement avec.

- *Partage*. Chaque musicien utilise la base de patterns crée par un autre. Ceci permet de jouer avec des patterns inhabituels : les patterns typiques de saxophone, piano ou guitare sont, en effet, assez différents, pour des raisons de morphologie d'instrument.

5. Résultats

Il est bien sûr difficile de décrire la musique et en particulier une improvisation. Le système présenté ici peut produire de longues phrases souvent impressionnantes, linéaires mais expressives, dans des styles guitaristiques divers (Pat Martino, Pat Metheny, Alan Holdsworth, John McLaughlin). Toute évaluation du système doit cependant être conduite non pas sur le système lui-même, mais sur le couple musicien/système. A cet égard, les écoutes réalisées devant différents auditeurs montrent que la plupart de ceux ci ne sont pas capables de distinguer les contributions humaines et de la machine. Par ailleurs, la qualité de l'improvisation (ici encore, difficile à quantifier) dépasse de loin, à notre avis, ce que produisent les systèmes autonomes. Laissons les auditeurs juger d'eux-mêmes : quelques enregistrements sont disponibles à www.csl.sony.fr/~pachet/music. Enfin, nous pensons que nos expérimentations permettent de relancer la question de l'improvisation dans un contexte à la fois plus réaliste (on passe du système autonome à l'instrument intelligent) et, finalement, plus humain.

References

Assayag, G. Dubnov, S. Delerue, O. (1999) Guessing the Composer's Mind: Applying Universal Prediction to Musical Style, Proc. ICMC 99, Beijing, China, I.C.M.A., San-Francisco.

Biles, John A. (1998) Interactive GenJam: Integrating Real-Time Performance with a Genetic Algorithm, Proc. ICMC 98, Ann Arbor, Michigan.

Chemillier, Marc (2001) Improvising Jazz Chord Sequences by Means of Formal Grammars, submitted to ICMC 2001, Cuba.

Heuser, Jorg (1994) Pat Martino – His contributions and influence to the history of modern Jazz guitar. Ph.D thesis, University of Mainz (Ge).

Johnson-Laird, P. N. (1991). "Jazz Improvisation: A Theory at the Computational level." Representing Musical Structures, P. Howell, R. West, and I. Cross, eds., Academic Press, 291-325.

Pachet, F. Surprising harmonies. *JIM 1999*, Issy-Les-Moulineaux.

Pachet, F. Qu'est ce qu'une mélodie intéressante. *La Recherche*, numéro spécial sur les origines de l'art, Novembre 2000.

Martino, Pat. (1993) Creative Force, Vol. 1 and 2, CPP/Beldwin, Miami.

Ramalho G., Ganascia J.-G. (1994) Simulating Creativity in Jazz Performance. Proceedings of the National Conference in Artificial Intelligence, pp. 108-113, AAAI-94, Seattle, AAAI Press.

Rolland, P.Y. 1999. Discovering Patterns in Musical Sequences. *Journal of New Music Research* 28:4, December 1999. Pages 334-350.

Wessel, David, Improvisation with Highly Interactive Real-Time Performance Systems, in Proc. Int. Computer Music Conf. (ICMC'91), pp. 344-347.

Ziv, J. Lempel, A. "Compression of individual sequences via variable rate coding", *IEEE Trans. Information Theory*, 24, (5), 1978.

Tiling the Line (Pavage de la ligne)
Self-Replicating Melodies, Rhythmic Canons, and an Open Problem

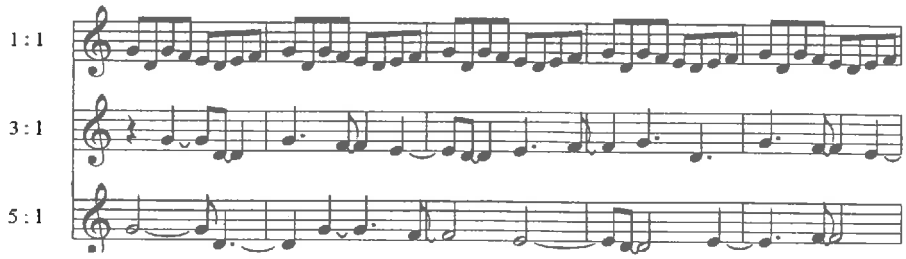
Tom Johnson

Summary: Melodic loops can be broken up into modules or tiles that are identical or multiples of one another. Recent investigations, following precedents in some 12-tone theory, and in the work of D. T. Vuza, reveal some new ways of doing this, as well as some new unanswered questions.

Tiling the plane (pavages en deux dimensions) has been studied ever since the Greeks demonstrated how to do this with regular polygons. Such information was so interesting for geometry, and so useful in architecture, mosaics, and all the "decorative" arts, that regular advancements have been made until today, when one can find large books cataloguing hundreds of tilings or tessellations.

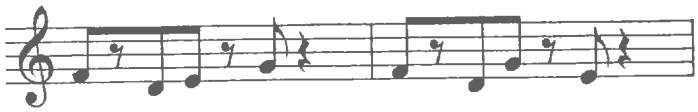
Tiling the line, however, which is just doing the same thing in one dimension, has been rarely treated, and almost exclusively as a musical problem. Probably the first serious studies were about tiling the chromatic scale with combinatorial hexachords and tetrachords and such, for composing 12-tone music. D. T. Vuza, taking the problem into the time dimension, advanced our knowledge beyond the cyclical group Z/12Z and studied "rhythmic canons" of many different lengths, but we still don't know very much, though it is beginning to seem that tiling the line may one day be almost as rich and complex a subject as tiling the plane. And for a minimalist like myself, who has nothing against repeating loops, and who seeks music with very tight logical organization, there is much potential here.

To begin, let me show you the loop used in the *Kientzy Loops*, written for saxophone and tape for Daniel Kientzy. It is an eight-note loop that constitutes a "self-replicating melody," making a copy of itself at 3 : 1 and also at 5 : 1.

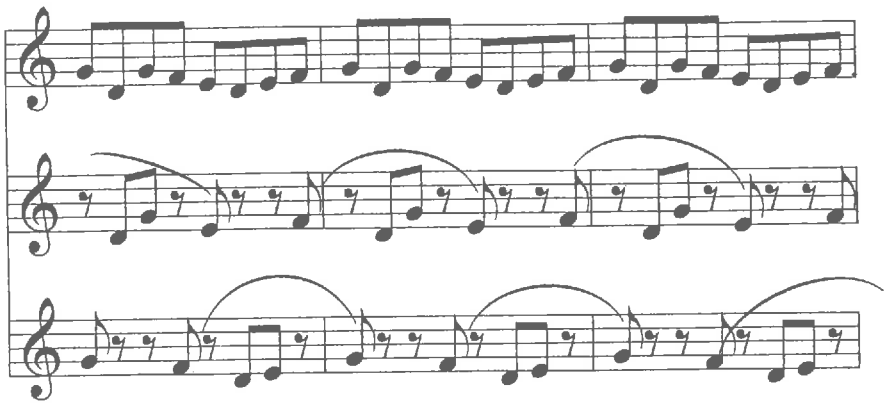


Since writing this piece a year ago, I have become interested in loops that not only make copies of themselves, but which also can be broken up into canons that tile the line. So I thought it would be a good exercise for me now to try to tile this eight-point line with two four-point modules. In other words, I want to construct the eight-note loop as a canon of two similar four-note sub-loops.

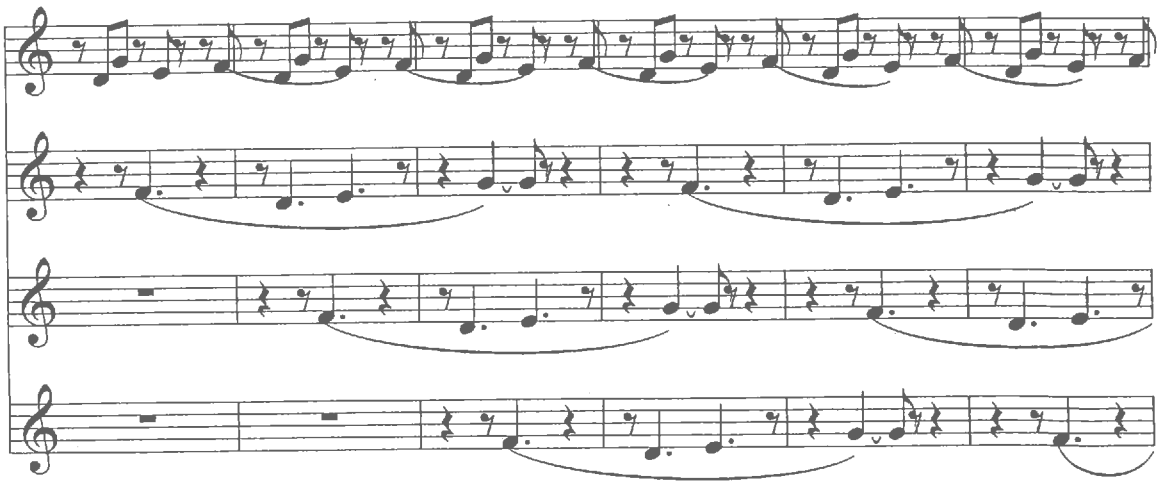
The loop contains two G's, two F's, two E's and two D's, so it seemed possible that some combination of the four pitches would be complemented by the other four pitches. I tried all the possibilities and found no way in which this loop can be divided into two four-note sub-loops that are exactly the same, but we can work with this combinatorial rhythm, where the two sub-loops are only slightly different:



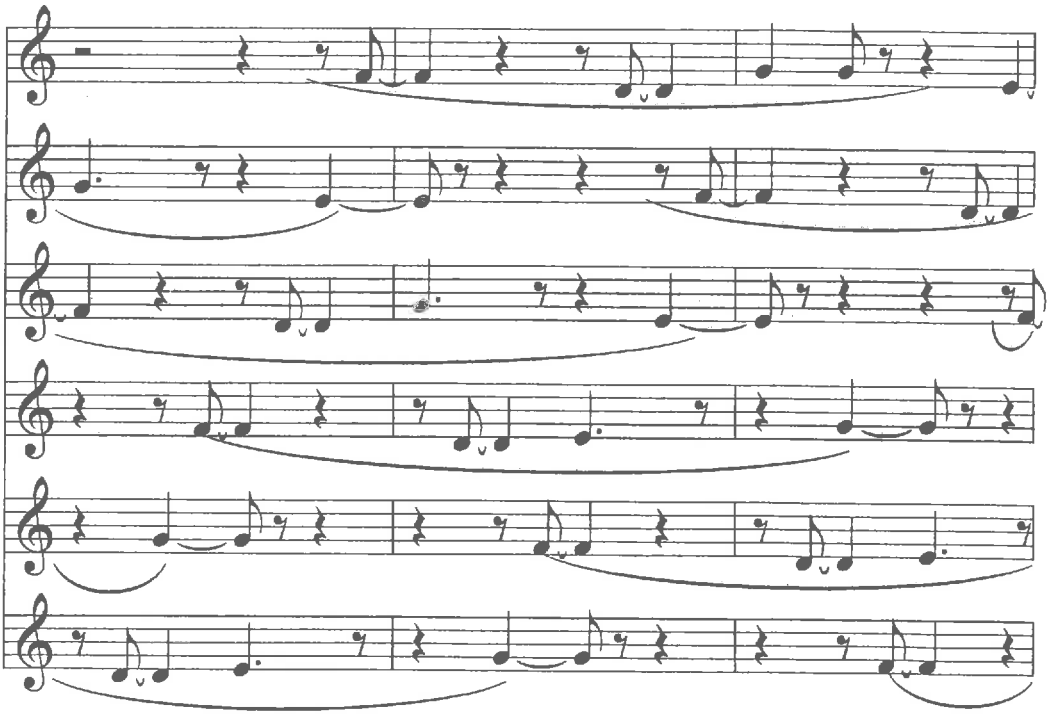
Now we can have a self-replicating eight-note loop divided into two four-note sub-loops in such a way that one can hear the complete loop and listen to an inexact two-voice canon at the same time.



Since the basic loop makes a copy of itself at 3 : 1 and 5 : 1, the sub-loops do too, so we can divide one of these loops into three loops, each moving three times slower than the basic 1 : 1 tempo of the other sub-loop, and have an inexact four-voice canon, with the voices in two different tempos. Now our tiling has some holes and some simultaneities, but the notes always occur at exactly the right moments within the original eight-note loop.



Or we can split both sub-loops into three voices and have a six-voice canon, with all the voices at one tempo:



Now the line is filled perfectly, with no simultaneities and no holes, and since the basic loop makes a copy of itself at 5 : 1, as well as 3 : 1, we could also transform our loop into an eight-voice inexact canon with three voices moving 3 times slower than the basic loop and five voices moving five times slower, or a 10-voice inexact canon, with all voices moving at 5 : 1. We could even complicate things further with voices moving 9 or 25 times slower than the basic eight-note pulse, but the general idea should already be clear.

When D. T. Vuza did his basic work on "rhythmic canons," he never considered polyrhythmic situations such as these, where the tiles covering the line are moving at different tempos. Thomas Noll, on the other hand, has taken much interest in these possibilities, and has a great facility for finding polyrhythmic solutions. When I asked him, for example, if it was possible to compose a 14-beat loop that would also be a rhythmic canon in which the voices would have tempos of 3 : 5, he responded immediately:

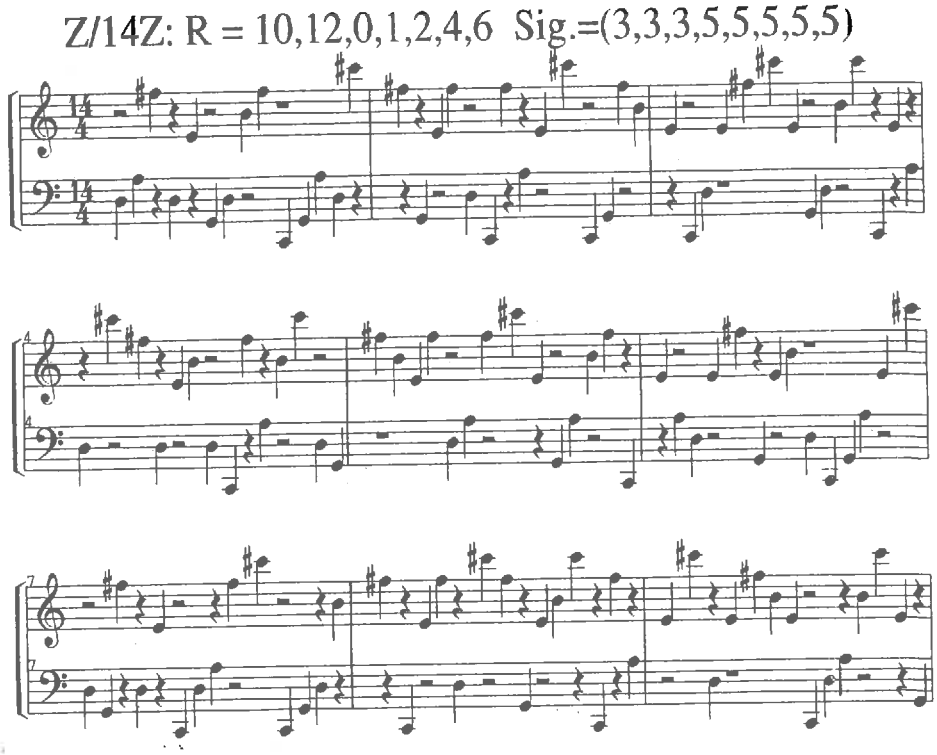
There is only one interesting solution:

The fundamental rhythm is R = {0, 1, 2, 4, 6, 10, 12}, it has several canons with signatures
{{1, 1}, {1, 3}, {1, 5}, {2, 2}, {2, 4}, {2, 6}, {3, 3}, {3, 5}, {4, 4}, {4, 6}, {5, 5}, {6, 6}}

This is the {3,5} canon:

{{0,3,6,12,4,2,8},{5,10,1,11,7,13,9}}

By "signature," he means the tempo ratios of the tiles employed. Here you can see Noll's solution in musical notation. Each of the eight pitchs represents one of the eight voices, three of which move at 3 : 1 and five of which move at 5 : 1:



This example was already mentioned in my lecture on *Objets mathématiques trouvés*, as part of the philosophy-mathematics-music seminar at IRCAM in January, a rather long paper, involving numerous esthetic points and many additional examples of loops, along with other findings of Thomas Noll, and some useful mathematical observations by Markus Reineke. That lecture is easy to find on line, so I won't say any more about that, and will continue here with a new problem, which remains completely open. Here is the way I defined it in correspondence with several colleagues earlier this spring:

I want to write a composition using a rhythmic canon with the rhythm (0,1,4), permitting three different tempos in the ratio of 4 : 2 : 1. Just as one must avoid holes and overlaps in traditional two-dimensional pavings, I want to avoid both simultaneities and empty beats in this composition. The problem is that I don't know whether there are a finite number of solutions, in which case it would be nice to somehow use them all, or whether the number of solutions is infinite, in which case I must find some other way to structure the piece.
Regarding this tiling problem as a game, our first move can be either the basic motif (0,1,4), or one of the two augmentations, multiplied by 2 and by 4, as indicated by the letter "a" here. We could also multiply by other numbers, but let's keep the problem on a manageable level for now:

aa00a (0,1,4)
a0a00000a (0,2,8)
a000a00000000000a (0,4,16)

Taking only the first possibility, there are three possible second moves, which I'll notate as b's:

aabba0b
aa0bab00000b
aab0a0b00000000000b
aa0ba00b00000000000b

Taking only the first of these possibilities, there are two possible third moves that fill up the hole, as shown with c's:

aabbacbc00000c
aabbacb000c00000000000c

Taking only the first of these possibilities, there are several possible fourth moves.

aabbacbcdd00dc
aabbacbc000ddc0d
aabbacbcd0d00c00d
aabbacbc0d0d0c000d
aabbacbc00d0dc0000d
aabbacbc0000dcd00000d
aabbacbcd000dc0000000000d
aabbacbc00d00cd00000000000d
aabbacbc000d0c0d00000000000d
aabbacbc0000dc00d00000000000d

Some of these are dead ends, because they leave behind holes that can never be filled with the tiles available, but some of these fourth moves could be continued with several possible fifth and sixth moves, and of course, there are many possible first, second, third and fourth moves that we have not followed up, so there are many possible tilings. But how many?

The situation is not as complex as a game of chess, which offers 20 possible first moves, but there are a lot of possibilities all the same. Simply counting all the possible solutions for a finite loop of 30 or 60 or 75, or some other number divisible by three, could take quite a bit of time.

Marc Chemillier responded that the problem was difficult, and that I should ask D. T. Vuza Vuza responded that the problem is extremely difficult, and that he is no longer working with musical questions. Maybe I will just find some interesting way of tiling some loop having a legth divisible with three and write the piece anyway, but wouldn't it be nicer if I could know that I was using all the possibilities, or that no one will ever be able to use them all?

References

Johnson, Tom: *Self-Similar Melodies*, 1996, 289 pp, Editions 75.
Johnson, Tom: *Self-Replicating Loops*, *JIM* 1999. Or *www.tom.johnson.org.*
Johnson, Tom: *Objets mathématiques trouvés, Seminaire Entretemps Musique, mathématiques et philosophie*, IRCAM, January 2001.
Vuza, D.T. *Supplementary Sets and Regular Complementary Unending Canons, Perspectives of New Music,* 1990-91

Outils d'analyse musicale et recherche sur le contenu : une démarche musicologique

Jérome Barthélemy, Alain Bonardi, Marcos Bezerra de Menezes
IRCAM
1, place Igor Stravinsky
75004 Paris France
33 01 44 78 48 43
jerome.barthelemy@ircam.fr, alain.bonardi@ircam.fr, marcos.bezerra@ircam.fr

Résumé

Nous présentons ici les outils d'analyse musicale que nous implémentons dans le cadre du projet européen WedelMusic de distribution électronique de partitions musicales, et comment ces outils servent à enrichir la base de données d'un catalogue local, afin de permettre une recherche sur le contenu. Après une brève description de l'architecture générale du projet Wedel actuellement en cours de développement, et des principes retenus pour la protection du droit des auteurs, nous décrivons les principes de base des procédés que nous mettons en œuvre, et leur justification du point de vue musicologique. Nous décrivons les algorithmes d'extraction de motifs caractéristiques, et les structures musicales sur lesquelles ils portent. Enfin, nous décrivons l'architecture de la base de données, dans la perspective de son extensibilité.

Mots clés : analyse musicale, recherche de motifs similaires, similarité, encodage musical

1. Présentation générale du projet WedelMusic

WedelMusic – pour Web Delivering of Music Scores – est un projet financé par la Communauté Européeenne. Il vise la distribution électronique de partitions musicales tout en protégeant les droits des auteurs et des éditeurs. Ce projet est en cours de développement et devrait s'achever en 2002. Il rassemble des partenaires de plusieurs pays, dont deux éditeurs de musique (Ricordi et Suvoni Zerboni en Italie), un laboratoire universitaire (Dipartimento di sistemi et informatica de l'Université de Florence en Italie), des centres de recherche (le Fraunhofer Institute for Computer Graphics en Allemagne, l'Ircam en France, Institute for Language and Speech Processing en Grèce) et d'enseignement (Ecole de Musique de Fiesole en Italie), des organismes institutionnels (Agenzia per l'alta tecnologia en Italie, Studie-en VakBibliotheek voor visuel en anderszins gehandicapten aux Pays-Bas) et des groupes privés (Artec Group en Europe).

Dans cette optique, les partitions musicales sont mises à disposition du public dans des lieux spécialisés, qui peuvent être des organismes privés ou publics, médiathèques, conservatoires, écoles de musique, librairies musicales etc. Un équipement spécifique, constitué d'une base de données – un catalogue – et d'un ensemble d'outils spécialisés disponibles sur des terminaux est installé dans ces locaux. L'ensemble de ces équipements constitue ce que nous appellerons ici un distributeur local.

L'administrateur du distributeur local constitue son catalogue à partir des catalogues disponibles chez les éditeurs. Le catalogue, ainsi que son contenu, est disponible sur des terminaux, sur un réseau local connecté au Distributeur Local, ainsi que différents outils pour visualiser la partition, pour l'analyser, la transposer, la réduire etc. Le catalogue est de même disponible sur Internet pour des interrogations, et éventuellement pour des recherches sur le

contenu, mais le contenu lui-même ne peut être visualisé, et ce pour des raisons de protection de droits.

1.1. Protection des droits

Dans le projet WedelMusic, une attention particulière est apportée à la protection des droits des auteurs et des éditeurs. Dans cette architecture, les fichiers Wedel – partitions, fichiers MIDI, documents audio – résident toujours sur le serveur sous une forme cryptée.

Pour chaque usage de la partition effectué à l'aide des outils Wedel, une demande d'autorisation est effectuée en ligne auprès du serveur de l'Editeur, et les droits afférents sont reportés auprès de l'administrateur du Distributeur Local. Après obtention de l'autorisation, les objets Wedel résidant dans la base de données du Distributeur Local sont décryptés sur les terminaux à l'aide d'une clé délivrée en ligne.

1.2. Les outils de recherche dans le catalogue

Les outils de recherche dans le catalogue sont basés d'une part sur les méta-données fournies par l'éditeur : titre de l'œuvre, compositeur, dates, classification sommaire, genre etc...., et d'autre part sur des motifs caractéristiques extraits de la partition sur la base des outils d'analyse implémentés sur les terminaux.

Dans le cadre du projet Wedel, il n'est pas possible d'implémenter les outils de recherche sur le contenu directement sur la partition, car celle ci doit résider sur le serveur sous forme cryptée, et ceci pour répondre aux exigences de protection des droits des auteurs. Il est donc nécessaire de passer par une phase d'extraction de motifs caractéristiques, qui fait l'objet d'une procédure en ligne d'autorisation auprès du serveur de l'éditeur.

Après l'obtention de cette autorisation, les motifs caractéristiques peuvent être stockés sur le serveur sous une forme non cryptée, et peuvent être rendus disponibles pour les outils de recherche sur le contenu.

2. Les principes de base de la détection de motif

La détection automatique de motif caractéristique dans une œuvre musicale pose en premier lieu le problème de la détection des articulations. Les critères que l'on peut envisager pour définir les divisions sont multiples : on pourrait par exemple retenir les pauses, les différences de registre, les différences de timbre, une combinaison de ceux-ci. Aucune de ces solutions ne semble réellement satisfaisante.

Nicolas Ruwet [Ruwet 1966], en étudiant la démarche du musicologue, constate que la méthode est rarement explicitée, aussi bien dans le domaine de la détection du mode que dans le domaine de la détection des divisions, périodes, phrases, membres, incises. Il écrit notamment : « Mais la question cruciale, préliminaire à toutes les autres, est la suivante : quels sont les critères qui, dans tel cas particulier, ont présidé à la division ? Or, cette question, personne ne prend la peine d'y répondre, comme si l'évidence des critères sautait aux yeux. »

Partant de cette constatation, Ruwet retient, comme critère principal de division, la répétition. L'importance de la répétition en musique, et à tous les niveaux, a par ailleurs déjà été abondamment illustrée, notamment par Schoenberg [Schoenberg 1947 ; Schoenberg 1950].

Un des intérêts de ce critère est par ailleurs son apparente universalité. Gilbert Rouget [Rouget 1961] l'applique aux musiques africaines : « certains fragments sont répétés, d'autres ne le sont pas ; c'est sur la répétition – ou l'absence de répétition – qu'est fondé notre découpage. Lorsqu'une suite de sons est énoncée à deux ou plusieurs reprises, avec ou sans variante, elle est considérée comme une unité. »

Il convient bien entendu de noter ici l'emploi du mot « variante ». La variante, ou la variation, joue un rôle aussi important en musique, et généralement dans les arts, au moins occidentaux, que la répétition dont elle est le corollaire quasiment obligé. Ainsi Schoenberg note en 1950 [Schoenberg 1950] : « la musique dans le style homophonique et mélodique, c'est à dire la musique avec un thème principal, accompagné et basé sur l'harmonie, produit son matériel à l'aide de ce que j'appelle le développement par variation. Cela signifie que les variations sur une unité de base produisent toutes les formulations thématiques qui apportent d'une part la fluidité, les contrastes, la variété, la logique et l'unité, et d'autre part le caractère, le mode, l'expression, et toute la différenciation requise, et donc l'idée de la pièce. »

Schoenberg cite maints exemples de ces processus de développement par variation, qu'il étudie notamment chez Brahms ou Mozart, ou bien dans sa propre musique [Schoenberg 1947].

3. Mise en pratique, encodage, modèle et algorithmique

Nous avons choisi d'étudier particulièrement les méthodes et algorithmes utilisés dans le domaine de la biologie, en ce qu'ils semblent poser et résoudre le même type de problèmes que ceux rencontrés dans la détection de motifs musicaux, c'est à dire détection des répétitions, et tolérance aux erreurs, ou aux variantes. Cette similitude a été remarquée et étudiée par de nombreux auteurs : cf par exemple [Crochemore 1998], [Cambouropoulos 1999].

Dans le cas particulier de la biologie, les variations, ou variantes, sont induites principalement par la présence de mutations dans la chaîne d'ADN, mutations qui doivent être révélées par la méthode. Nous examinerons plus précisément, en essayant de les appliquer sur des cas concrets, quelques-unes des méthodes couramment utilisées dans le cas de la biologie. Une bonne revue d'ensemble de ces méthodes peut être trouvée dans [Gusfield 1997].

La méthode de la différence lexicale consiste à mesurer la distance entre deux unités lexicales, à l'aide d'une méthode telle que la distance de Hamming, ou la distance « des pâtés de maison »[1], et à décider de la similarité en se basant sur un seuil , - ou un maximum - à ne pas dépasser.

Nous nous intéresserons aussi au procédé d'alignement de chaînes. Ce procédé consiste à aligner deux chaînes de caractères en y insérant des blancs, de manière à ce que chaque caractère dans l'une des chaînes se trouve placé vis à vis du même caractère dans l'autre chaîne, ou d'un blanc.

Un autre procédé consiste à mesurer la « distance d'édition », c'est à dire l'ensemble des opérations – insertion, suppression, remplacement - qui doivent être effectuées sur une chaîne

[1] La distance de Hamming mesure le nombre d'indices pour lesquels les deux séquences diffèrent. La distance des « pâtés de maison » est la somme des valeurs absolues des différences.

de caractères pour obtenir la seconde. Ce procédé nous apparaît comme particulièrement intéressant pour ses applications possibles dans le domaine musical.

3.1. Quelques cas

3.1.1. Distance lexicale

Nous examinerons tout d'abord l'application de la distance lexicale sur un simple exemple issu d'une *Invention* de Bach en Fa Majeur :

Figure 1. *Les premières mesures de l'Invention en Fa Majeur de J.-S. Bach.*

Les deux premières occurrences du motif en croches peuvent être encodées par intervalles diatoniques, (indépendamment du qualificatif mineur/majeur/juste), pour les rendre indépendants de la transposition :

x : 3, -3, 5, -5, 8
y : 3, -3, 6, -4, 6
delta : 0, 0, 1, 1, -2
Distance de Hamming : 3
Distance des « pâtés de maison » : 4

Voici maintenant les deux motifs encodés par intervalles (avec le qualificatif mineur/majeur/juste) :

x : 3M, -3M, 5J, -5J, 8J
y : 3m, -3m, 6m, -4J, 6M
Distance de Hamming : 5

Dans ce cas, aucune correspondance n'est trouvée.
Nous pouvons aussi supposer un encodage dans lequel les notes seraient encodées à l'aide de l'intervalle qui les sépare de la première note :
x : 3, 1, 5, 1, 8
y : 3, 1, 6, 3, 8
delta : 0, 0, 1, 2, 0
Distance de Hamming : 2
Distance des « pâtés de maison » : 3

Dans ce cas précis, trois correspondances sont trouvées.

Cet exemple simple met en lumière les difficultés rencontrées en utilisant ce type de méthode. Il pose les questions de savoir comment calculer un indicateur, et comment calculer une limite pour décider si deux motifs sont similaires.

Si nous appliquons la distance de Hamming, nous trouvons dans le premier cas une distance de 3, dans le second une distance de 5, et dans la troisième une distance de 2.

En appliquant la distance des « pâtés de maison », nous trouvons dans le premier cas une distance de 4, dans le troisième une distance de 3 (n'ayant pas établi de règle de mesure de différence dans le deuxième cas, nous ne pouvons pas calculer cette distance).

D'autre part, cela nous permet d'introduire une remarque complémentaire : l'évaluation de similarité basée sur la distance lexicale est fortement liée au choix de la méthode d'encodage.

3.1.2. Alignement de chaînes

Le concept d'alignement de chaînes intéresse le musicologue dans la mesure où il sait que le compositeur procède souvent par insertion de notes supplémentaires, processus connu sous le nom d'ornementation, ou bien au contraire par suppression de notes.

Nous examinerons brièvement les premières mesure de la *Sonate en la mineur* pour piano de Mozart:

Figure 2. *Le début de la Sonate en La mineur de Mozart.*

Le second élément, y, est dérivé de x par insertion de notes supplémentaires. Le troisième élément z est dérivé du second par réduction : suppression de la désinence et ornementation. Voici une présentation des trois éléments montrant le résultat de la procédure d'alignement.:

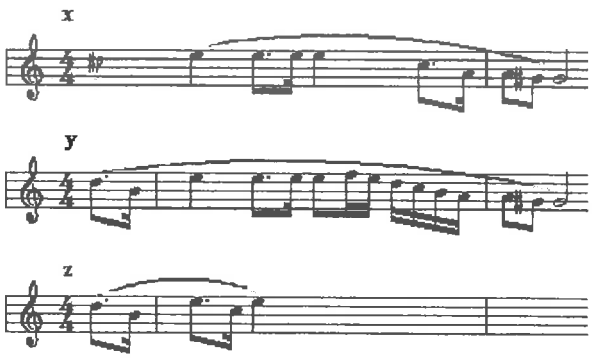


Figure 3. *les trois éléments montrant le résultat de la procédure d'alignement.*

Nous pouvons observer que les notes alignées occupent la même position rythmique, à l'exception notable du Do dans les deux premiers motifs. Nous pouvons aussi observer que la durée de ce Do dans le premier élément est telle que le Do dans le second élément tombe à l'intérieur de cette durée. Cette observation ne surprendra pas le musicologue habitué à la pratique des retards et des appoggiatures dans la musique des XVIII^{e} et XIX^{e} siècles.

Cette observation peut être utilisée pour renforcer la similarité entre les deux motifs, et pourrait être exprimée de la manière suivante : un alignement est trouvé lorsque les notes alignées occupent la même position à l'intérieur de la mesure, ou bien que la position de l'une est comprise entre la position de l'autre et de celle qui la suit.

L'application de cette règle montre que le troisième élément peut être aligné sur le second qui peut lui-même être aligné sur le premier.

Le problème est maintenant de choisir un encodage correct pour montrer ces alignements.
Si nous choisissons un encodage basé sur les intervalles, nous ne trouverons pas les alignements. Nous pouvons revenir à un encodage basé sur les notes, ou bien choisir une méthode dans laquelle les intervalles sont encodés par rapport à une note de référence (ici la tonique). Nous obtenons l'alignement suivant :
x : -, -, 5, 5, 5, 5, -, -, -, 3, -, 1, 1, 7, 7
y : 4, 2, 5, 5, 5, 5, 6, 5, 4, 3, 2, 1, 1, 7, 7

3.1.3. Distance d'édition

Nous allons aborder cette méthode avec un exemple tiré du second mouvement du *Sextuor* de Brahms opus 18.

Le motif des 8 premières mesures, exposé à l'alto, est composé de trois variantes du motif initial de 2 mesures, composé lui-même d'un intervalle de quarte ascendant en anacrouse, suivi d'un accent sur le temps principal, et d'une désinence en forme de seconde ascendante, avec une ornementation très simple formée d'une simple broderie. Ce motif est répété deux fois, et modifié par augmentation de l'intervalle initial successivement en sixte, puis en octave, et par insertion d'une ornementation complexe pour remplacer la simple broderie.
Ce motif est présenté ci-dessous de manière à montrer le résultat d'un alignement :

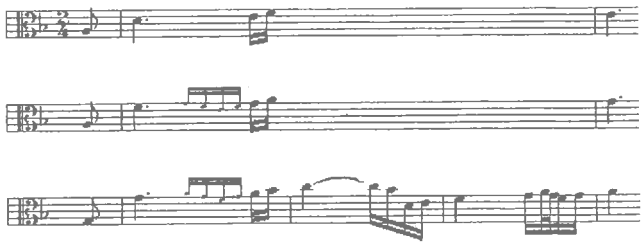


Figure 4. *Le thème principal du second mouvement du Sextuor de Brahms opus 18.*

Malheureusement, il est impossible d'appliquer strictement ici une procédure d'alignement, du fait de la modification de l'intervalle initial. De plus, le choix d'un encodage basé sur les intervalles rendrait l'alignement des deux dernières notes impossible. Le choix d'un encodage basé sur les notes rendrait l'alignement impossible du fait de la transposition.

Nous avons choisi une représentation par intervalles pour étudier la distance d'édition, dans la quelle les intervalles sont codés avec leur valeur, suivi d'un + ou d'un – pour indiquer la direction (dans cet encodage, l'ornementation explicitement écrite en « petites notes » est délibérément ignorée) :
Premier élément : 4+, 2+, 2+, 2-
Second élément : 6+, 2+, 2+, 2-
Troisième élément : 8+, 2+, 2+, 2+, 2-, 6-, 2+, 2+, 2+, 2+, 2-, 2-, 2+, 2+

Pour obtenir la transcription de la distance d'édition, nous choisissons d'écrire un A lorsqu'un intervalle est augmenté, un D lorsqu'un intervalle est réduit, un 0 lorsqu'il n'y a pas de modification, un – lorsqu'un intervalle change de direction, et un I pour une ou plusieurs insertions successives (nous ignorons délibérément ici l'ornementation explicitement écrite en petites notes).

La transcription d'édition du premier au second élément : A, 0, 0, 0
La transcription d'édition du second au troisième élément : A, 0, 0, –, I
Nous pouvons considérer que ces transcriptions sont de « bonnes transcriptions », et peuvent être stockées en tant que telles pour servir de références ultérieures (bien entendu, il s'agit là d'une décision prise par le musicologue, et non par un processus automatique).

3.2. Choix de la méthode d'encodage, et introduction des descripteurs.

Nous avons constaté à plusieurs reprises que le choix de la méthode d'encodage influençait fortement les résultats des procédures de comparaison. Cette remarque a déjà été faite par le passé, et notamment par P.-Y. Rolland dans sa thèse de doctorat « Découverte automatique de régularité dans les séquences et application à l'analyse musicale » [Rolland 1998]. Nous décrivons ici deux descripteurs particuliers, le descripteur de contour en ce qu'il peut fournir une base pour la détection

3.2.1. Descripteur de contour

Cette méthode encode chaque intervalle avec un c pour un intervalle conjoint, un d pour un intervalle disjoint, avec un + ou un – indiquant la direction. Une note répétée est encodée avec un =. Appliquée à l'Invention de J.-S.Bach ci dessus, cette méthode donnera le descripteur suivant : +d, -d, +d, -d, +d, -c, -c, -c, +c, -c, -c, -c, +c, -c, -c, -c, +d, +d, -d, +d, -d, +d etc... Une simple procédure de détection des répétitions exactes révèle immédiatement la paire maximale[2] suivante : +d, -d, +d, -d, +d

Si nous appliquons cette méthode à un corpus de quelque importance, nous obtiendrons probablement une avalanche de résultats, dû au fait qu'il n'existe que 3125 combinaisons de 5 éléments différents employant un alphabet de 5 caractères. Ces résultats peuvent être validés - ou invalidés - par des procédures complémentaires de comparaison appliqués sur des descripteurs de plus haut niveau , par exemple par calcul de la distance lexicale, ou bien par comparaison de la transcription de la distance d'édition avec une transcription de référence, ou bien par comparaison de la structure rythmique.

3.2.2. Descripteur de mélodie et position rythmique

Dans l'exemple de la *Sonate* de Mozart étudié ci-dessus, nous avons vu que l'encodage correct pour appliquer un alignement devait être basé sur les notes et non sur les intervalles. Ce type d'encodage est dépendant de la transposition, ce qui signifie que si une transposition intervient, nous ne pouvons pas détecter d'alignement, sinon en effectuant des calculs complexes qui vont aboutir à un algorithme inefficace.

Une solution à ce problème pourrait être trouvée en encodant plusieurs fois la même mélodie à l'aide des intervalles la séparant d'une note de référence, et en choisissant une note différente à chaque fois. Nous ajoutons aussi un encodage correspondant à la position de la note à l'intérieur de la mesure (par rapport à l'unité de la double croche).

[2] Une paire maximale est une paire de chaînes telles que les éléments placés immédiatement à la droite et à la gauche de la première sont différents des éléments placés à la droite et à la gauche des éléments placés immédiatement à la droite et à la gauche de la seconde.

Appliqué au *Sextuor* de Brahms, cet encodage donne les résultats suivants :
Avec le Ré comme note de référence :
5:12, 1:0, 2:8, 3:12, 2:0, 5:12, 3:0, 4:8, 5:12, 4:0, 4:12, 4:0, 5:8, 6:12, 7:0, 6:10, 1:12, 2:14, 3:0, 4:8, 5:10, 4:11, 3:12, 4:14, 5:0
Avec le Sol comme note de référence:
2:12, 5:0, 6:8, 7:12, 6:0, 2:12, 7:0, 1:8, 2:12, 1:0, 1:12, 1:0, 2:8, 3:12, 4:0, 3:10, 5:12, 6:14, 7:0, 1:8, 2:10, 1:11, 7:12, 1:14, 2:0
Les éléments alignés sont soulignés.

Il convient de constater que l'intervalle initial n'est pas inclus dans cet alignement, c'est dire que cette procédure ne nous semble pas encore satisfaisante pour détecter cette similarité.

3.2.3. Formalisation des descripteurs

Avant de décrire plus précisément le modèle mis en oeuvre, nous donnons ici quelques précisions sur la formalisation des descripteurs tels que nous les mettons en œuvre.
Chaque descripteur doit définir les éléments suivants :

- Un identifiant de type, unique.
- L'alphabet (le jeu de caractères), et l'élément vide (un élément qui ne peut jamais apparaître dans une chaîne)
- Le "facteur de perte d'information" , ou, a contrario, le facteur de précision.
- Les dépendances (les descripteurs dont il est dérivé).
- Les méthodes qui peuvent lui être appliquées.
- La taille minimum d'un élément.

3.3. Modèle

L'ensemble de ces constatations nous amène à développer un modèle dans lequel la recherche des motifs caractéristiques est effectuée en plusieurs temps :
Premièrement, une recherche de répétition exactes dans ce que nous appelons des descripteurs « de bas niveau » (par exemple, le descripteur de contour défini ci-dessus). Deuxièmement, une phase de vérification - consolidation des motifs trouvés, par recherche de motifs similaires dans les descripteurs « de haut niveau ».

Cette phase est elle même divisée en deux phases :

- une première phase de « vérification » : pour chaque paire trouvée, une évaluation de similarité est effectuée sur des descripteurs différents (par application des procédures de différence lexicale ou de distance d'édition). Les paires non vérifiées sont rejetées.
- une deuxième phase de « consolidation » dans laquelle les motifs retenus (les différentes instances des paires) font l'objet d'une recherche exhaustive de motifs similaires, par application des procédures d'alignement (et utilisation des algorithmes de programmation dynamique). Cette phase doit permettre de détecter des paires qui n'auraient pas été détectées par la recherche de répétitions exactes.

Une troisième phase dans laquelle les motifs retenus sont triés en fonction de leur longueur et du nombre de leurs occurrences.

3.4. Algorithmique

3.4.1. Détection des répétitions exactes

Dans le cadre de la biologie, les algorithmes les plus couramment utilisés pour la détection de répétitions exactes sont basés sur l'arbre des suffixes. Des solutions ont été pour la construction de l'arbre des suffixes avec un temps d'exécution linéaire, cf Weiner [Weiner 1973] et Ukkonen [Ukkonen 1995].

Toutefois, cet objectif est difficile à réaliser en pratique, et n'est strictement exact que pour une taille d'alphabet constante, en raison de la taille mémoire nécessaire pour l'exécution de l'algorithme. Les implémentations courantes de cette méthode sont des compromis entre le temps d'exécution et l'espace mémoire nécessaire.
Nous avons donc implémenté dans un premier temps une méthode de force brutale, dont le temps d'exécution est en n^2. Nous avons implémenté dans un deuxième temps une méthode dont le temps d'exécution moyen est en n log n.

3.4.2. Détection des motifs similaires

Dans le cas de la détection des motifs similaires, nous développons une méthode basée sur la programmation dynamique.

3.4.3. Application à la détection de structures similaires

Dans le cadre du projet Wedel, nous développons aussi un processus de « réduction » de la partition. Cette réduction comprend tout d'abord une réduction au sens harmonique du terme, qui comprend elle-même plusieurs phases.

La première phase est composée d'une quantification de chaque mesure par unité rythmique minimale, et d'une agrégation verticale de toutes les voix dans chaque unité rythmique. Ces agrégats verticaux ainsi obtenus sont ensuite agrégés entre eux, d'abord par temps, puis par mesure, par application d'un corpus de règles issues de la théorie de l'harmonie, qui permettent de décider si deux agrégats peuvent être de nouveau agrégés ou non. Dans ce processus, certaines notes sont éliminées en fonction de règles mélodiques et harmoniques, qui permet d'éliminer les « notes étrangères ».

La dernière phase est une détection de la basse : est considérée comme « basse » la note la plus basse d'un accord qui n'appartient pas à l'accord précédent.

L'ensemble de ces procédures génère un descripteur qui peut être considéré comme composé d'une basse et d'un chiffrage, auquel les procédures de détection de répétition décrites ci-dessus peuvent être appliquées.

De même, un processus de reconnaissance de tonalité est développé, fondé sur un corpus de règles mélodiques et harmoniques. Ce processus génère un descripteur de tonalités, sur lequel les procédures de détection de répétition peuvent être appliquées.

De plus, un descripteur dérivé du descripteur « harmonie » peut être généré, dans lequel la basse peut être exprimée sous forme de degrés (par rapport à la tonique), et un processus de détection de structures similaires peut être implémenté.

4. Architecture de la base de données

Afin de favoriser l'échange de données, et l'extensibilité, l'architecture de la base de données est entièrement calquée sur un formalisme XML, c'est à dire que pour chaque partition, une notice XML descriptive peut être extraite de la base. De même, une notice XML peut être utilisée pour alimenter la base de données.
Cette équivalence est fondée sur trois tables :

- une table des conteneurs,
- une table des attributs,
- une table des contenus.

La table de conteneurs présentée à la figure 5 reproduit la structure de l'arborescence XML (à chaque conteneur correspond un tag XML).
Chaque conteneur possède:

- un type,
- un identifiant unique (container_id) dans la base de données,
- un conteneur parent (sauf pour les conteneurs "racine").

TABLE DES CONTENEURS : tbl_containers		
type (chaîne de caractères)	**parent_id** (entier)	**container_id** (entier)
les types peuvent être: ROOT (conteneur "racine") thematicelement descriptor tonality KeySignature TimeSignature nomenclature Instrument [...]	identifiant du conteneur parent (les conteneurs de type ROOT n'ont pas de parent_id)	identifiant unique du conteneur dans la base de données

Figure 5. *Table des conteneurs.*

Remarque : le conteneur racine possède des attributs permettant d'identifier l'objet correspondant et de relier ainsi le contenu aux méta-données fournies par l'éditeur (elles-mêmes inscrites dans la base de données), et au contenu multimédia de la base.

La table des attributs (figure 6) regroupe les paires clé/valeur pour chaque tag XML.
Chaque paire clé/valeur est reliée au conteneur par son identifiant unique (container_id).

TABLE DES ATTRIBUTS : tbl_attributes		
att_key (chaîne de caractères)	**att-value** (chaîne de caractères)	**container_id** (entier)
contient les clés	contient les valeurs	identifiant unique du conteneur qui contient la paire clé/valeur. un conteneur peut avoir plusieurs paires clé/valeur

Figure 6. *Table des attributs.*

La table des contenus (figure 7) regroupe les différents contenus pour chaque tag XML.
Chaque contenu est relié au conteneur par son identifiant unique (container_id).

TABLE DES CONTENUS : tbl_contents	
content (chaîne de caractères)	**container _id** (entier)
contenus.	identifiant unique du conteneur correspondant. un seul contenu par conteneur.

Figure 7. *Table des contenus.*

Une procédure simple, basée sur une API en C++, permet d'extraire une notice XML à partir de la base de données, en spécifiant l'identifiant du conteneur racine.
De même, en sens inverse, une procédure permet de lire une notice XML et d'importer le contenu correspondant dans la base de données.

5 Conclusion

Dans cet article, nous avons justifié et décrit un modèle d'extraction de motif caractéristique fondé sur une approche musicologique, voire ethnomusicologique. Le modèle que nous avons décrit est actuellement en cours d'implémentation, et devrait entrer dans une phase expérimentale de validation durant l'année 2002.

Nous avons d'autre part bien conscience des limitations imposées par l'exercice : si la répétition semble bien être un critère assez universel, la notion de variante par contre est fortement liée à une culture, car il semble bien que ce soient des critères culturels qui nous permettent de reconnaître deux motifs comme étant apparentés l'un à l'autre.

Une évolution nécessaire du système devrait pouvoir permettre au musicologue de définir lui-même les règles d'élaboration de nouveaux descripteurs, lui permettant ainsi d'élaborer sur des exemples concrets des schémas d'interprétation, qu'il pourrait ensuite étendre à des corpus plus larges. De cette manière, il pourrait mettre en œuvre l'analyse à double sens, « top-down » et « bottom-up » telle que la souhaitent Pierre Boulez [Boulez 1989], et Alain Bonardi [Bonardi 2000].

Références

[Bonardi 2000] BONARDI, A., Music IR for Contemporary Music : What the Musicologist Needs, Proceedings of the first International Symposium on Music Information Retrieval, Plymouth (USA), october 2000.

[Boulez 1989] BOULEZ, P., *Jalons (pour une décennie) – Dix ans d'enseignement au Collège de France*, Paris, Christian Bourgois Editeur, 1989, 451 p.

[Crochemore 1998] CROCHEMORE, M., ILIOPOULOS, C. S., YU, H., *Algorithms for Computing Evolutionary Chains in Molecular and Musical Sequence* in the Proceedings of the ninth Australian Workshop on Combinatorial Algorithms AWOCA'98, C.S. Iliopoulos, ed., School of Computing, Curtin University of Technology, Perth, Western Australia, 1998, p 172 - 184.

[Cambouropoulos 1999] CAMBOUROPOULOS, E., CROCHEMORE, M., ILIOPOULOS, C.S., MOUCHARD, L. and PINZON, Y.J. (1999), *Algorithms for Computing Approximate Repetitions in Musical Sequences*, in the Proceedings of the AWOCA'99 Workshop Australasian Workshop on Combinatorial Algorithm 25-27 July 1999, Perth, Western Australia.

[Gusfield 1997] GUSFIELD, D., *Algorithms on strings, trees, and sequences*, Cambridge, Cambridge University Press, 1997, 534 p.

[Rolland 1998] ROLLAND, P.-Y., *Découverte automatique de régularités dans les séquences et application à l'analyse musicale*, Thèse de doctorat, UNIVERSITÉ PARIS 6, Juillet 1998, 335 p.

[Rouget 1961] ROUGET, G., *Un chromatisme africain*, l'Homme, I. 3, 1961, p.41.

[Ruwet 1966] RUWET, N., *Méthodes d'analyse en musicologie*, in *Revue belge de Musicologie*, 20 (1966), p 65-90.

[Schoenberg 1947] SCHOENBERG, A., *Brahms the progressive* , in *Style and Idea* , Faber and Faber, 1975, p. 398-441.

[Schoenberg 1950] SCHOENBERG, A., *Bach*, in *Style and Idea*, Faber and Faber, 1975, p. 393-397.

[Ukkonen 1995] UKKONEN, E., *On-line construction of suffix trees*, Algorithmica, 14:249-60, 1995.

[Weiner 1973] WEINER, P, *Linear pattern matching algorithms*, Proceedings of the 14th IEEE symposiun on switching and automata theory, 1973.

TRAM, un outil générique d'analyse formelle appliqué à la musique.

Benoit Meudic
Ircam, équipe représentations musicales
1. place Igor Stravinski 75004 Paris
meudic@ircam.fr

Résumé : Cet article présente une librairie d'analyse (TRAM) développée dans l'environnement Open-Music [Agon 98] de l'Ircam. Cette librairie se base sur une théorie de la forme, la "théorie du rythme" [LUSSON 86] qui a été conçue par un mathématicien et musicologue Français, Pierre Lusson.
Développée en CLOS (Common Lisp Object System), la librairie offre un ensemble d'outils d'analyse et de description de la forme destinés aussi bien à l'analyse, à la composition musicale à la génération textuelle [Lusson - Roubaud 2001] ou encore à la "retrouvaille d'informations musicales" (ou RIM).

Mots clefs : Théorie de la forme, analyse musicale, génération musicale.

1- Présentation de la théorie :

Ces dernières années, on a pu constater une demande croissante pour des théories musicales pouvant se traduire en algorithmes [Leman 97].
Parmi celles qui sont apparues, la "Generative Theory of Tonal Music" [Lerdhal et Jackendoff 83] est l'une des plus connues. Elle utilise un ensemble de notions fondamentales comme le groupement, la répétition de groupement (la métrique) ou la hiérarchisation, appliquées à des séquences d'évènements.
Ces notions sont appliquées dans un contexte particulier (la musique tonale) et le plus souvent sur un type de séquence particulier (une séquence de notes ou d'accords).

La théorie du rythme prend en compte ces notions, mais elle se place à un niveau de généralité plus élevé. Elle offre une sorte de meta-langage qui permet non seulement d'opérer des analyses sur n'importe quel type d'objet séquentiel, mais aussi de manipuler les outils d'analyse dans un contexte abstrait qui constitue une sorte de pont entre les différents domaines d'application (musique, texte, séquence d'images). Pour cela, elle utilise de nouvelles notions.
Par exemple, elle introduit la notion d'éminence (voir la définition plus loin) et celle de niveau (voir définition plus loin). La notion d'événement élémentaire (voir définition plus loin) qui se confond avec celle de note dans la théorie de Lerdhal et Jackendoff devient beaucoup plus générale.

Pour résumer, les principales caractéristiques que veut avoir la théorie sont :
(A l'heure actuelle, ces caractéristiques ne sont pas toutes entièrement formalisées, certaines constituent des objectifs que de futures recherches essaieront d'atteindre)

1. Son extrême niveau de généralité qui consiste à analyser un objet décrit par les seules notions abstraites de frontière, d'éminence et de mêmeté (voir la définition dans la partie suivante).
Ceci explique que la théorie puisse être appliquée aussi bien à la musique qu'au texte ou à la séquence d'images.

2. Son entière formalisation ce qui autorise le développement d'algorithmes d'analyse et de génération. La théorie peut ainsi être appliquée au domaine de l'informatique musicale, par exemple pour la recherche de descripteurs musicaux de haut niveau.

3. Son indépendance vis à vis des considérations esthétiques ou stylistiques, ce qui la rapproche de la théorie de Narbour [Narmour 1990]. Pour respecter cette indépendance, la difficulté est de mettre en évidence dans des cas précis les paramètres et les axiomes particuliers qui constitueront la théorie esthétique de ce style.

4. Son indépendance vis à vis des hypothèses cognitivistes ou perceptives sur lesquelles on voudrait fonder l'analyse. Au contraire, l'analyse permet de dégager des concepts cognitifs ou perceptifs qui peuvent ne pas coïncider avec les concepts habituels [Fraisse et Deliège].

2- Contenu de la théorie :

Dans cette partie on fera une brève description des objets fondamentaux de la théorie.

Considérations générales :

La théorie du rythme ne s'intéresse qu'aux objets séquentiels. Les objets non séquentiels (une image) ne peuvent pas être analysés par la théorie.
La théorie comprend deux parties distinctes :
Une partie abstraite (tram), qui s'intéresse à une combinatoire des groupements d'éléments discrets.
Une réalisation (trr, théorie du rythme réalisé), qui s'intéresse aux mêmes objets que la tram, mais en réalisant ces objets dans les différents domaines d'application (musique, texte, rapport texte-musique, neurophysiologie cognitiviste).
Trois notions fondamentales sont à la base de la théorie : les notions de frontière (de groupement), d'éminence et de mêmeté.

Nous allons définir les principales notions utilisées dans l'analyse d'une séquence d'objets.
Pour commencer, la théorie considère les objets à analyser comme des suites d'évènements élémentaires :

Les évènements élémentaires (é-é) :
Ce sont les briques de base qui seront manipulées par la théorie.
Pour les obtenir, on opère une segmentation de l'objet à analyser (un texte ou un fichier midi).
On obtient une séquence d'évènements élémentaires auxquels seront attribués des propriétés (par exemple, en musique, les propriétés peuvent être la durée, la dynamique, ou la hauteur…).
Tous les é-é pourront vérifier ou ne pas vérifier ces propriétés. Ainsi, pour une propriété choisie, ils pourront être comparés entre eux.

Le niveau :
Le niveau est en quelque sorte le grain de définition auquel on se place en choisissant l'é-é. Par exemple, pour un texte, on peut choisir de se placer au niveau du mot, de la syllabe, de la phrase. Un é-é sera respectivement un mot, une syllabe ou une phrase.

Sur ces séquences d'évènements élémentaires, la théorie opère des marquages :

Les marquages :
Les différents é-é peuvent être comparés sous l'aspect du même et du différent pour chaque propriété considérée. En effet, chaque propriété est ou n'est pas vérifiée pour chaque é-é, ce qui permet d'établir deux catégories d'é-é par propriété.
Si une propriété est vérifiée, on attribue un poids de 1 et 0 sinon.
Un marquage pour une propriété donnée est l'association de la séquence d'é-é avec la liste des poids 0 ou 1.

Des marquages se dégage la notion d'éminence :

Eminence :
L'éminence d'un é-é est la somme des éminences (suites de poids) relatives à chacun des marquages considérés pour cet é-é. Les éminences relatives d'un marquage peuvent être pondérées par un coefficient de manière à déterminer l'importance du marquage par rapport aux autres marquages considérés. Pour un système de marquages, la séquence d'éminences obtenue par combinaison linéaire des marquages sera appelée « mélodie de poids ».
Ainsi, les marquages : (0 1 0 1 0 1 0 1) et (1 0 1 0 1 0 1 0), lorsqu'ils sont combinés par l'addition, avec un coefficient 1 pour le premier et -1 pour le second, forment la mélodie de poids suivante : (-1 1 –1 1 –1 1 –1 1).

En fonction des éminences, on opère des groupements :

Groupement :
Les maximaux locaux d'une suite de poids correspondent aux é-é vérifiant localement le plus de propriétés. Ce sont des points d'articulation autour desquels on peut effectuer des groupements d'é-é.

Frontières :
Les frontières sont les séparations entre les groupements d'é-é d'une séquence.

Des notions plus complexes peuvent alors être considérées :

Mêmeté :
Dans le cadre des groupements, une première question est celle de leur combinatoire, ce qui implique immédiatement le traitement de la répétition. Cette répétition doit être considérée au sens large, on utilise pour cela la notion de mêmeté. Dans le cas le plus général, deux groupements seront dits « les mêmes » lorsqu'ils présenteront les mêmes variations dans la suite de poids.

Mètre :
Le mètre généralise cette notion de groupement.

Un mètre est une liste d'éléments qui pour au moins un niveau est concaténation d'un même groupement, dit générateur.
Par exemple, la séquence ((. .)(. .)(. .)) est un mètre.

3- Description de la librairie Open-Music

La librairie Open-Music comporte trois parties :

- un ensemble de fonctions d'analyse de texte
- un ensemble de fonctions d'analyse musicale
- un ensemble de fonctions génériques qui intègrent les notions abstraites de la tram (théorie du rythme abstrait mathématique) définies plus haut.

Les fonctions d'analyse de texte et d'analyse musicale effectuent une analyse préalable (analyse syntaxique, phonétique ...pour le texte, harmonique, rythmique... pour la musique) de l'objet pour disposer d'informations qui permettront d'effectuer une segmentation en é-é.

Cette analyse préalable dépend du contexte dans lequel on se trouve (musique classique, sérielle, poème, prose...).
L'analyse peut être automatique, mais le type d'analyse à effectuer est choisi par l'utilisateur suivant l'aspect de l'objet qu'il veut privilégier pour l'établissement des futurs marquages.

Ensuite à l'analyse préalable, les é-é seront associés à des propriétés en vue d'une manipulation par les marquages.
Les autres fonctions d'analyse sont des fonctions de marquage. Chaque marquage a pour sortie une liste de poids.
Par exemple, les marquages musicaux prennent en compte les paramètres midi habituels (hauteur, onset, durée, dynamique) en considérant le contour ou en les considérant par rapport à un seuil. Des marquages plus complexes prennent en compte les propriétés d'une séquence tonale en mesurant par exemple la distance entre les degrés des accords et le premier degré de la tonalité.

La troisième partie de la librairie est consacrée à la tram (théorie du rythme abstrait mathématique).
Ces fonctions réalisent les opérations de combinaison linéaire des marquages, de changement de niveau, de groupement, d'inversion, de comparaison par mêmeté, de comparaison par référence à un mètre. Je ne les détaillerai pas dans cet article.

4- Exemples

Exemple 1

La librairie TRAM est utilisée dans un algorithme de détection de pulsation (beat tracking) qui a été développé dans la librairie KANT d'Open-Music [Agon 94].
Ce modèle est assez proche de celui établi par Simon Dixon [Dixon & Cambouropoulos 2000].

Par analogie avec les domaine des hauteurs, la pulsation est considérée comme étant l'association d'une phase et d'une fréquence. Le modèle comprend donc deux parties : la première que l'on peut qualifier d'induction établit plusieurs hypothèses sur la fréquence (ou pulsation), puis dans une deuxième partie la séquence d'éléments (une suite de notes, d'accords, de durées) est parcourue afin de déterminer les éléments placés sur l'occurrence d'une des fréquences émises en hypothèse dans la première partie (recherche de la phase).

La théorie du rythme est utilisée dans chaque partie.
Pour cela, une analyse de la séquence d'éléments est effectuée en considérant deux marquages. Pour l'instant, seuls les inter-onset (i-o) entre les éléments de la séquence ont été considérés.
Le premier marquage marque chaque élément dont l'i-o est supérieur à celui de l'élément précédent. Le poids attribué est proportionnel au nombre d'éléments précédents d'i-o inférieur.
Le deuxième marquage marque chaque élément dont l'i-o est similaire à celui de l'élément précédent.
Les distances séparant chaque maximum de la mélodie de poids somme des deux marquages sont interprétées comme autant d'hypothèses sur la pulsation pour la première partie.
Dans la deuxième partie (recherche de la phase), les évènements de poids maximal seront préférés à leurs voisins en cas d'ambiguïté si plusieurs sont candidats à une même occurrence de pulsation. De cette manière, les variations de tempo ne pertuberont pas l'algorithme.

Exemple 2 :

La librairie peut être utilisée pour établir des rapports entre un texte et une musique. Pour cela, le texte et la musique sont tous deux analysés à l'aide des fonctions de la librairie : établissement des marquages, pondération des marquages, puis calcul de la mélodie de poids. La comparaison des deux mélodies de poids obtenues révèle le rapport qu'entretiennent le texte et la musique.
Ce type d'analyse a été utilisé par le compositeur Francois Sarhan pour mettre en musique le poème "la rue tombe noire" de Jacques Roubaud, à la différence qu'au lieu d'analyser la musique, il généré la musique en utilisant la contrainte des marquages musicaux mis en relations avec les marquages du texte.

Voici un exemple d'analyse d'un fragment de texte et de sa mise en musique :

Exemple d'analyse d'un fragment de "lobsters quadrille" de Gÿorgi Ligeti :

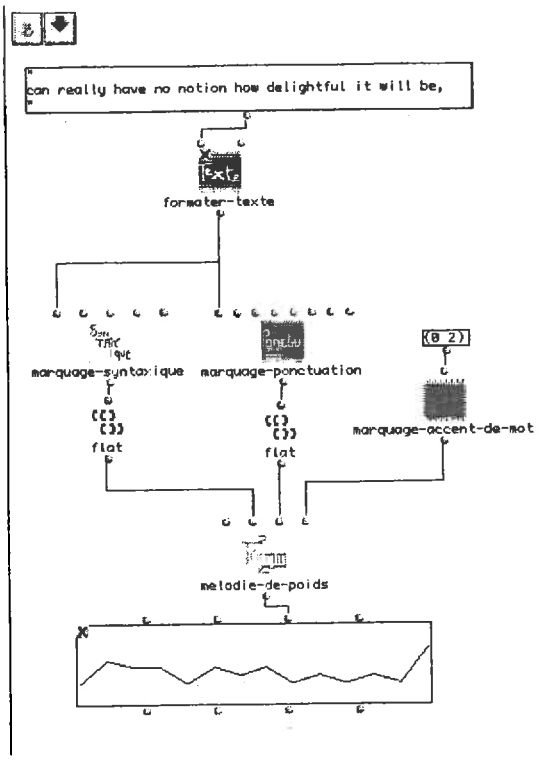


Fig 1

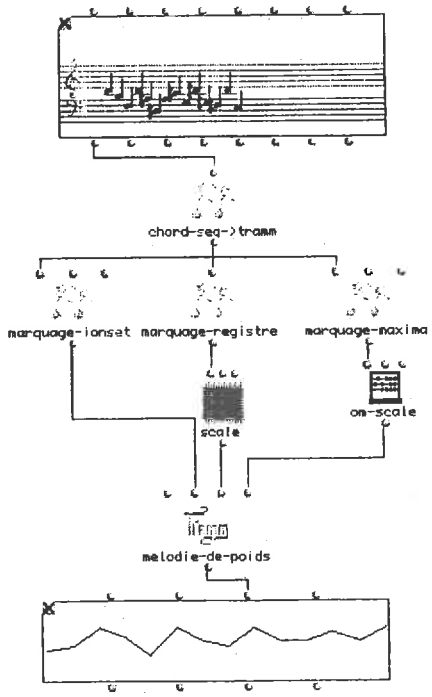


Fig 2

Fig 1 :
Analyse dans Open-Music d'une phrase du texte de Lewis Carroll.
Les calculs se font de haut en bas.
La fonction formater-texte recherche les propriétés du texte dans le dictionnaire. Elle transmet le résultat aux fonctions de marquage (marquage syntaxique, de ponctuation, et d'accentuation) qui sont appliqués au texte segmenté par syllabes. La mélodie de poids est calculée (somme des trois marquages). Elle apparaît sous forme de graphe.

Fig 2 :
Même analyse que précédemment, mais cette fois-ci à partir de la phrase musicale.
Les marquages utilisés sont : marquage d'inter-onset (donne un poids proportionnel à la valeur de l'inter-onset), de maximum mélodique global (marque la plus grande variation de hauteur) et de registre (marque les notes de trois manières différentes suivant le registre de leur hauteurs).

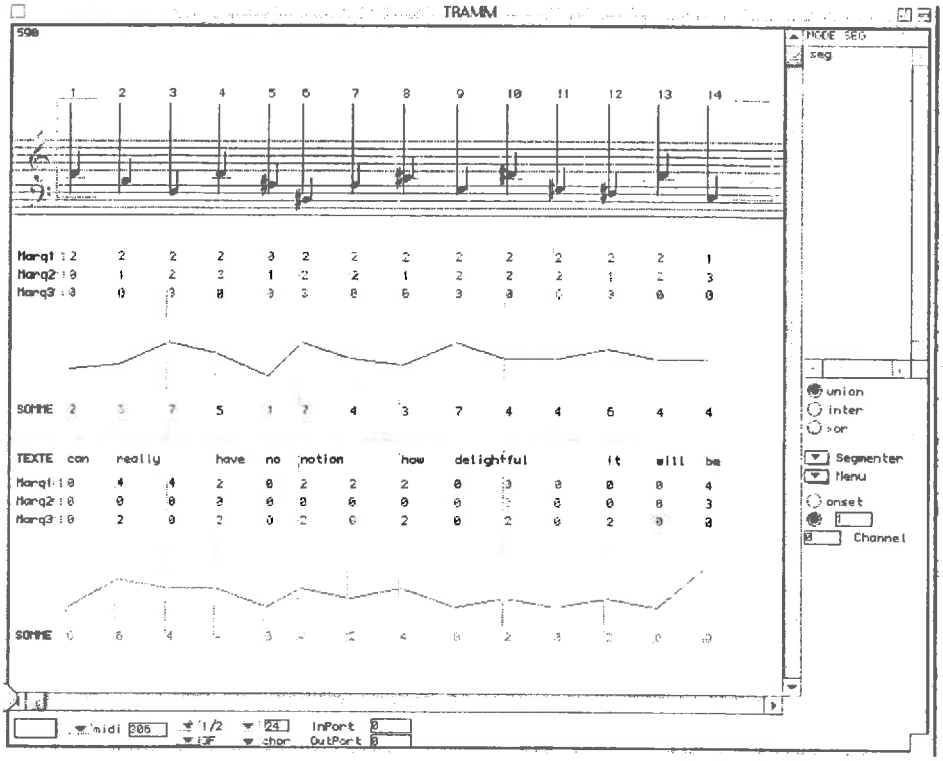


Fig 3

Fig 3 :

Les deux mélodies de poids sont copmparées dans un même fenêtre. Elles présentent un contour (variation haut, bas ou constant) proche sauf à deux endroits (really et how) ou le maximum de la mélodie de poids du texte est en avance par rapport à celle de la musique (effet de syncope).

5- CONCLUSION ET TRAVAUX FUTURS

Nous avons présenté une nouvelle librairie d'analyse musicale basée sur la théorie du rythme. Cette librairie fournit à l'utilisateur un ensemble de fonctions permettant l'analyse de la structure d'une surface musicale ou textuelle. Des exemples ont été présentés de manière à illustrer les différents champs d'application de la théorie. Maintenant, un travail devra être fait pour proposer des marquages musicaux plus pertinents adaptés à un contexte musical donné. Pour cela, l'étape d'analyse préalable devra être affinée de façon à obtenir un plus grand nombre de propriétés complexes. Par exemple, l'analyse harmonique ou l'extraction des voix d'un fichier midi ne sont pas encore réalisées (une analyse harmonique est implémentée mais n'est efficace que pour des structures harmoniques sans ambiguïtés). De même que pour l'exemple de détection de pulsation, cette analyse préalable pourrait être effectuée à l'aide de la théorie du rythme. Ainsi, par la théorie, des propriétés sont extraites à un certain niveau, puis nourrissent le niveau au-dessus qui peut à son tour être analysé par la théorie.

6- REFERENCES

[Agon 94]
Kant, a critique of pure quantification
Preceedings of the ICMC 94, Aarhus

[Agon 98]
Openmusic, un langage visuel pour la composition musicale assistée par ordinateur.
Thesis, Ircam 1998

[Benzécri 1955]
Cours de linguistique de rennes

[Dixon & Cambouropoulos 2000]
Beat tracking with musical knowledge.
ECAI, 2000

[Leman 97]
Music, Gestalt and computing.
Springer-Verlag, berlin, 1997.

[Lerdhal Jackendoff 83]
A generative theory of tonal music.
Cambridge : MIT press,1983

[LUSSON 86]
Place d'une théorie générale du rythme parmi les théories analytiques contemporaines
Analyse Musicale n° 2, pp. 44-51, 1986.

[Natiez 75]
Fondements d'une sémiologie de la musique
Union générale d'éditions, Paris, 1975

[Narmour 1990]
The analysis and cognition of basic melodic structures : the implication-realisation model.
University of chicago press.

[Lusson - Roubaud 2001]
Formes du temps in « Meslanges »

Knowledge and Learning Based Segmentation and Recognition of Rhythm Using Fuzzy-Prolog

Tillman Weyde

Research Department of Music und Media Technology

University of Osnabrück

Osnabrück, Germany

tweyde@uos.de

Abstract This paper introduces an architecture for rhythm recognition and comparative analysis. A fuzzy system is used to rate segmentation and structural assignment produced by combinatorial pattern-matching. The fuzzy system can be trained by examples. It provides fault tolerant, context sensitive and adaptive recognition of musical rhythm with a description of temporal and structural deviations.

1 Introduction

Fuzzy logic allows for the modeling systems with uncertain and incomplete knowledge. Fuzzy systems can be robust and show graceful degradation on their limits while their function is still interpretable, it is not just a black box. So it is surprising that the use of fuzzy logic is rare in the field of music theory and musical applications[1], since our knowledge of processes involved in musical activity is far from being exact or complete.

The immediate recognition of auditory rhythmic structures in one of the key features of human auditory perception, necessary for understanding speech as well as music. Although there is a large body of research on the properties of auditory perception of temporal patterns[2], a coherent paradigm or theoretical framework to support computer models and applications is not yet established. This is a problem for musical computer applications like production tools, tutorial programs an musical database applications which should interact with the user in a musically meaningful way.

This paper introduces a framework for modeling the recognition of musical rhythmic structure. It provides a basis for integration of prior knowledge from empirical experiments and

[1] There are a few examples like [Kostek, 1999] who uses fuzzy logic and there are some custom system designs like [Schottstead, 1989] which contain elements similar to fuzzy systems.

[2] E.g. [Deutsch, 1986], [Handel, 1989], [Desain and Windsor, 2000].

music theory with a learning by example approach.

1.1 Musical pattern processing

Musically meaningful structures are constrained by the auditory perception of music since the main form of consuming and appreciating music is listening (rather than reading). So the integration of properties of aural perception is obligatory. Also knowledge from music theory can be integrated (which is dependent on cultural context).

There are two main aspects of rhythmic structure which need to be considered for recognition and analysis: *segmentation* and *similarity*. The segmentation process divides a sequence of events into groups. These groups correspond to musical motifs which are combined to higher level patterns like musical phrases. The similarity of groups determines the internal structure of these patterns or the relation to a known model (e.g. a given rhythm which should be played by a student). Groups of notes are assigned to groups in a model or within a piece of music on the basis of similarity. Segmentation and assignment processes are highly interdependent and both depend strongly on context.

2 System architecture

The initial motivation for this work was to bring more musical intelligence into music tutorial applications. The user should try to play rhythms after hearing or reading them. Our aim is to give differentiated and musically meaningful responses to the user about how her or his input differs from the task and what she or he should take care of. This is the more difficult the larger the differences between task and input are since it is not clear which part of the input corresponds to which part of the task. Missing or extra notes or groups, wrong order of groups, changing tempo and timing deviations can only be noticed by the system if it recognizes which groups were played by the user, even if they are temporally or structurally distorted or misplaced.

Our system currently supports three modes: segmentation only, matching patterns and matching structures. The segmentation mode emulates segmentation by a listener. It can be seen as a partial implementation of the grouping rules by [Lerdahl and Jackendoff, 1983] restricted to the rhythm domain. The pattern matching mode emulates recognition of a single rhythmic group. It assigns the notes in the matched patterns and gives thus detailed information on the differences between task and input groups. The structure matching mode combines and extends the former two. It assigns the groups and allows a detection and description of structural changes like omissions, insertions or changed order of groups.

The general scheme is to combinatorially generate segmentations, assignments of groups, and assignments of notes and filtering them. Filtering reduces complexity by removing perceptually implausible interpretations. Features are extracted from segmentations and assignments and a rating is calculated by the Fuzzy-Prolog program. The alternative rated best determines system output. A Fuzzy-Prolog program can be viewed as a neural net, where rules and facts correspond to network nodes. So by node we mean just a different view on a fuzzy rule or fact.

2.1 Combinatorial processing

Input data

Input data represents musical notes based on MIDI data. The input events have three values that we use: onset time, duration, and key velocity (loudness). If there is a task to which the input is to be compared to, the task is encoded in the same way.

Segmentation and group assignments

There are perceptual constraints to the length and the duration of perceptual groups. The number of events in a group is restricted as is well known since [Miller, 1956]. The maximal number is a adjustable parameter in our system and a setting of 4 or 5 has shown to be adequate which agrees with the literature ([Handel and Todd, 1981], [Swain, 1986]). We model these constraints by generating all segmentations within the given range of group lengths and filtering out those that do not meet the constraints.

Empirical evidence suggests that durations of perceptual groups lie in a range of approximately 0.5 to 2 seconds ([Seifert et al., 1995]). It is also known that temporal proximity of events plays a role, relatively long distances between events tend to end a group ([Handel, 1973]). Since grouping by temporal proximity is dominant over accent grouping ([Deutsch, 1986]) we can filter out segmentations that grossly contradict grouping by proximity. We also assume that groups containing only one element should not occur unless they have considerable distance to the neighbor notes ([Lerdahl and Jackendoff, 1983]) else they are filtered out.

On the structure level input groups are assigned to task groups based on input and task segmentation. All possible assignments for all combinations of segmentations are calculated. Currently we do not filter group assignments.

Note assignments and tempo variants

In a match of input and task groups notes of the input are assigned to notes of the task or marked as additional (leaving the temporal relations of the other notes intact) or inserted (the rest of the group being moved by the amount of time occupied by note), task notes are marked as assigned, subtracted, and removed respectively.

For every assigned input group tempo variants are calculated based on pairs of assigned notes as anchor points. Different possibilities for the two anchor notes yield different tempo variants. The process is visualized in figure 1. The result allows measuring the deviation of the group from the expected position, the tempo deviation and deviations of the individual notes concerning timing and loudness. Only the tempo variant which produces the best similarity rating is used for further calculation in order to reduce calculation time.

2.2 Feature extraction

On both the group level and on the structure level features are extracted that form the basis for the similarity and segmentation rating.

For segmentations we calculate ratings based on their length and number of notes. For

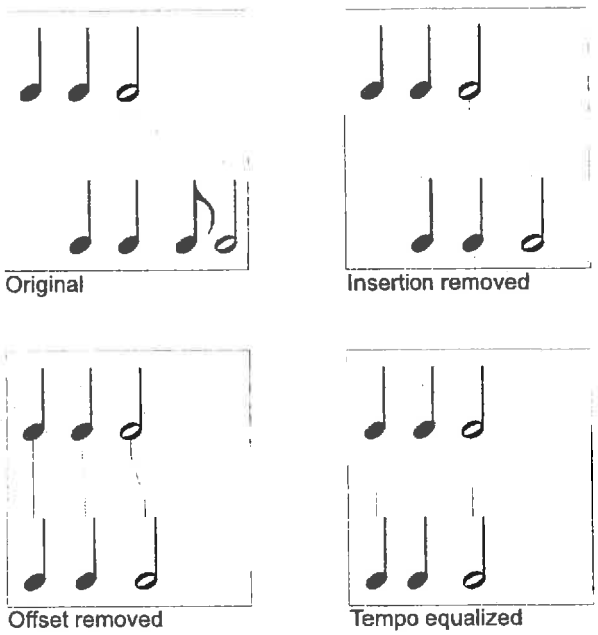


Figure 1: Calculation of tempo variants

length values we use four input nodes corresponding to different length. This design allows good fitting to a preference curve on the considered range of group lengths. The same design is used for the number of notes in a group. We have five different input nodes corresponding to note numbers from one to five. The regularity of group lengths (number of notes) as well as of group intervals (distance between the beginning of groups) is also rated by calculating their variance. For each group we have a node representing whether the inter-onset-interval (IOI) between the last note and the next group is larger than IOIs within the group, and whether the first note is relatively loud.

For similarity on the group and structure level nodes representing different kinds of imprecisions and inaccuracies are used as facts for the Fuzzy-Prolog program. We use nodes for early notes, late notes, too loud notes, too soft notes, added notes, inserted notes, subtracted notes, deleted notes, tempo stability, and tempo plausibility. The calculation of their values is not described here due to space limitations[3].

A feature specific for the structure level is the order of groups which is calculated as

$$\frac{i}{n \cdot (n-1)/2}$$

where i is the number of the intersections in the assignment graph and n the number of group assignments (see figure 2). The other nodes on structure level are calculated as combinations of the corresponding nodes per group.

[3]Some more details can be found in [Weyde, 2000] and [Enders and Weyde, 1996]

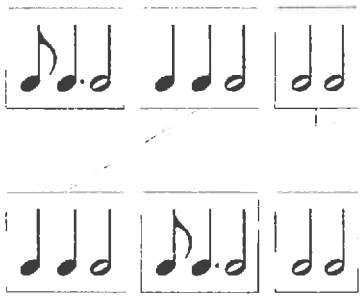


Figure 2: Group assignments for a rhythmic structure

3 Segmentation and similarity ratings using Fuzzy-Prolog

3.1 Fuzzy-Prolog

The module for rating segmentation quality and group similarity is based on *Fuzzy-Prolog* which was described by [Nauck et al., 1996]. Fuzzy-Prolog is a programming language similar to *Prolog*. A Fuzzy-Prolog program consists of a set of rules in the following form:

$$\text{conclusion} \leftarrow \text{premise} \wedge \ldots \wedge \text{premise}$$

Operators have an evaluation function with values in the interval $[0, 1]$. For evaluation of the implication the *Goguen-Implication* is being used. The truth value of other rules can be derived from those by using the *modus ponens* generalized for fuzzy logic [4].

3.2 Operators

We have conjunction and disjunction operators. There are several options for the evaluation function of disjuction and conjunction, well known ones are *min* for $\wedge$ and *max* for $\vee$. Since we need a differentiable function for learning by backpropagation we use this evaluation functions for conjunction:

$$f_{\wedge}(\|\varphi_1\|, \ldots, \|\varphi_n\|) = \left(\frac{\sqrt[q]{\|\varphi_1\|} + \ldots + \sqrt[q]{\|\varphi_j\|}}{n} \right)^q$$

and analogously for the disjunction:

$$f_{\vee}(\|\varphi_1\|, \ldots, \|\varphi_n\|) = \sqrt[q]{\frac{\|\varphi_1\|^q + \ldots + \|\varphi_j\|^q}{n}}$$

where $\|\varphi_n\|$ denotes the truth value of rule φ_n. These functions allow for a certain amount of compensation between the operands that can be adjusted by the q parameter. For these functions we use a q value of 2. For $q \to \infty$, $f_{\wedge}$ approaches *min* and $f_{\vee}$ approaches *max*.

[4] See [Nauck et al., 1996].

3.3 Fuzzy rules

Not all the rules that we use can be shown here due to space limitations. But we will look at an example to try and give an idea of how the system works. The rules for similarity on the structure level are based among others on the segmentation rating for input and task if present. E.g. the predicate for the input, CInputSegmentation, returns the quality of an input segmentation based on rules and facts on group and structure level:

$$\text{CInputSegmentation}(gs) \leftarrow \text{CInputSegment}(gs) \wedge \text{CEqualGroupDistance}(gs) \wedge \text{CEqualGroupLength}(gs) \quad (1)$$

$$\text{CInputSegment}(gs) \leftarrow \text{GInputSegment}(g_1) \wedge \ldots \wedge \text{GInputSegment}(g_n), \text{ where } gs = [c_1, \ldots, c_n] \quad (2)$$

$$\text{GInputSegment}(g_i) \leftarrow \text{GGroupDuration}(g_i) \wedge \text{GGroupLenth}(g_i) \wedge \text{GGroupEndLong}(g_i) \wedge \text{GGroupStartLoud}(g_i) \quad (3)$$

$$\text{GGroupDuration}(g_i) \leftarrow \text{GGroupDuration05}(g_i) \vee \text{GGroupDuration10}(g_i) \vee \text{GGroupDuration15}(g_i) \vee \text{GGroupDuration20}(g_i) \quad (4)$$

$$\text{GGroupLength}(g_i) \leftarrow \text{GGroupLength1}(g_i) \vee \text{GGroupLength2}(g_i) \vee \text{GGroupLength3}(g_i) \vee \text{GGroupLength4}(g_i) \vee \text{GGroupLength5}(g_i) \quad (5)$$

The facts of the form GGroupDuration*XX* (rule 4) are the inputs features mentioned earlier which return values close to one if the length of the respective group is near to 0.5, 1.0, 1.5 or 2.0 seconds. Similarly the rules GGroupLength*X* (rule 5) represent a group length of 1, 2, 3, 4, and 5. Relatively loud events tend to mark a group beginning and long events tend to end a group which is reflected by facts on group level (rule 3)

We do not know in advance how many groups there will be in a structure. The number of groups varies for different segmentations of the same input. This made it necessary to extend Fuzzy-Prolog with a list processing feature. In rule 2 the transition from the structure level to the group level takes place, where a variable number of inputs is combined to one node. We call these nodes with variable numbers of connections multi-nodes.

The output node CInterpretationQual returns the rating for a whole structure assignment that comprises segmentation, similarity of groups, and assignment of groups. The stucture of the net generated from the whole Fuzzy-Prolog program is shown in figure 3.

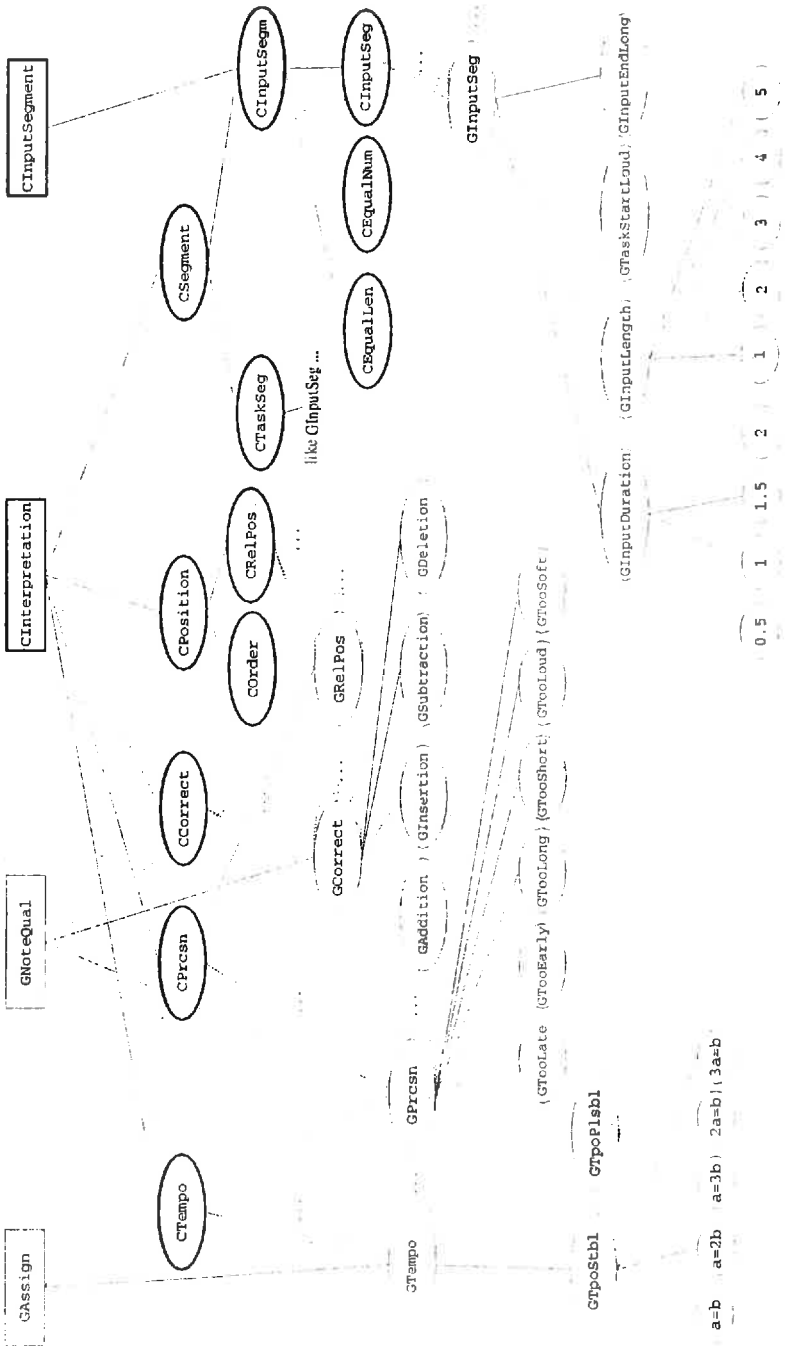


Figure 3: Structure of the neuro-fuzzy program system: Nodes on the top are output nodes (rectangles), Nodes on the bottom are net inputs Fuzzy-Prolog facts. Nodes with bold border operate on the structure level, all other nodes operate on the group level. Nodes with connections marked with '···' are transitions from group to structure level.

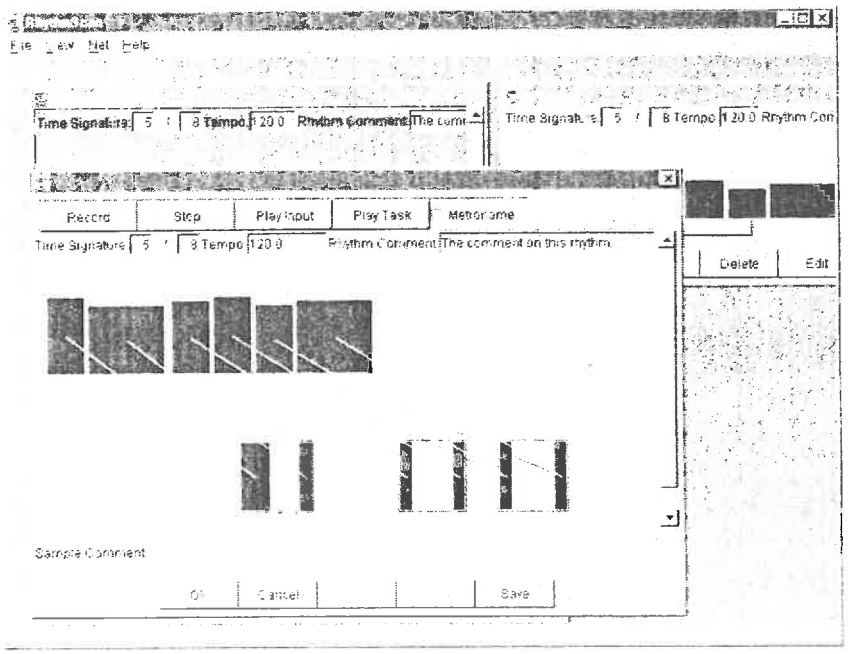


Figure 4: *RhythmScan* user interface: black lines show segmentation and assignments calculated by the system, yellow lines show input by an expert user

3.4 Learning

When trying to model how the rules contribute to the output in a fuzzy system, the rules must be weighed individually. These weights can be ad hoc estimates, but they have to be adjusted to achieve good performance. The idea is now to automate this process by optimizing the weights using rating examples. It is shown in [Nauck et al., 1996] that Fuzzy-Prolog programs are equivalent to feed-forward neural nets. They can therefore be trained with a version of the backpropagation algorithm adapted to the evaluation functions of the fuzzy logical operators. For the multi-nodes weight-sharing is used.

The system is trained with relative ratings following the method described by [Braun, 1997]. This means pairs of assignment or segmentation examples are given and one of each pair is to be rated better. When learning was successful the system prefers better assignments since they receive higher ratings.

4 Application

We have implemented an experimental application of the system described in this paper called *RhythmScan* which allows to generate samples, test, and train the system. It is a Java application with a Fuzzy-Prolog interpreter written in C and uses XML data format. The system can be interactively tested and trained by giving preferred segmentations and assignments. The training samples consist of a system output and the result preferred by the (expert) user. Differing segmentations and assignments by system and user can be seen in figure 4.

After training the system reacts adequately to segmentation tasks e.g. modeling subjective rhythmization and to perform structural comparison of user input for a given task like in interactive music tutorials. The system is able to detect delayed inputs, interrupted and repeated input as well as tempo and timing deviations.

5 Conclusions

Fuzzy-Prolog is a powerful means of modeling musical knowledge in combination with machine learning. The weights in the system trained by examples give also insight to the relevance of rules, although the sample sets have been to small to draw any generalizable conclusions yet. The system can be easily modified or extended by adding, changing or removing rules or input features.

The system works well on selected tasks but it has to be tested more intensively and systematically to explore the system capabilities. Further challenges are the integration of pitch and metrical information into the system as well as the extension to streaming mode and optimization of the computational performance.

References

[Braun, 1997] Braun, H. (1997). *Neuronale Netze*. Springer, Berlin Heidelberg.

[Desain and Windsor, 2000] Desain, P. and Windsor, L., editors (2000). *Rhythm Perception and Production*. Swets and Zeitlinger, Lisse.

[Deutsch, 1986] Deutsch, D. (1986). Auditory pattern recognition. In Boff, K. R., Kaufman, L., and Thomas, J. P., editors, *Handbook of Perception and Human Performance: Cognitive Processes and Performance*, volume 2, chapter 32, pages 32–1–49. John Wiley and Sons, New York.

[Enders and Weyde, 1996] Enders, B. and Weyde, T. (1996). Automatische Rhytmuserkennung und -vergleich mit Hilfe von Fuzzy-Logik. *Systematische Musikwissenschaft*, IV(1-2):101–113.

[Handel, 1973] Handel, S. (1973). Temporal segmentation of repeating auditory patterns. *Journal of Experimental Psychology*, 101:46–54.

[Handel, 1989] Handel, S. (1989). *Listening: An Introduction to the Perception of Auditory Events*. MIT Press, Cambridge, Massachusetts.

[Handel and Todd, 1981] Handel, S. and Todd, P. (1981). Segmentation of sequential patterns. *Journal of Experimental Psychology: Human Perception and Performance*, 7(1):41–55.

[Kostek, 1999] Kostek, B. (1999). *Soft computing in acoustics : applications of neural networks, fuzzy logic and rough sets to musical acoustics*, volume 30 of *Studies in fuzziness and soft computing*. Physica-Verlag, Heidelberg.

[Lerdahl and Jackendoff, 1983] Lerdahl, F. and Jackendoff, R. (1983). *A Generative Theory of Tonal Music*. The MIT Press, Cambridge, Mass.

[Miller, 1956] Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63(2):81–97.

[Nauck et al., 1996] Nauck, D., Klawonn, F., and Kruse, R. (1996). *Neuronale Netze und Fuzzy-Systeme*. Computational Intelligence. Vieweg, Braunschweig, 2 edition.

[Schottstead, 1989] Schottstead, W. (1989). Automatic counterpoint. In Mathews, M. V. and Pierce, J. R., editors, *Current Directions in Computer Music Research*, volume 2 of *System Develpment Foundation Benchmark Series*, chapter 16, pages 199–213. The MIT Press, Cambridge, Mass.

[Seifert et al., 1995] Seifert, U., Olk, F., and Schneider, A. (1995). On rhythm perception: Theoretical issues, empirical findings. *Journal of New Music Research*, 24(2):164–95.

[Swain, 1986] Swain, J. P. (1986). The need for limits in hierarchichal theories of music. *Music Perception*, 4(1):121–148.

[Weyde, 2000] Weyde, T. (2000). Recognition of rhythmic structure with a neuro-fuzzy-system. In Woods, C., Luck, G. B., Brochard, R., Seddon, F., and Sloboda, J. A., editors, *Proceedings of the Sixth International Conference on Music Perception and Cognition*, pages 1467–77, Keele, Staffordshire, UK. Department of Psychology, Keele University. CD-ROM (pdf, html).

Evaluating melodies by the complexity of polyrhythm

Andranik Tangian
FernUniversität Hagen, D-58084 Hagen
Andranik.Tangian@FernUni-Hagen.De

Abstract

A melody is considered as polyphonic, that is, composed by several latent voices. The rhythm structure of the latent voices constitutes the polyrhythm, whose complexity is estimated. It is supposed that 'good' melodies have some 'optimal' complexity which is not too high and not to low. The model is intended for computer composition and music education.

Keywords: Melody, rhythm, polyphony, polyrhythm, complexity.

1 Introduction

Inventing a good melody is generally considered to be the most creative and the least rationalizable task of music composition. If one asks "What is a good melody?" there is very little to answer. A good melody is easier to remember, or at least easier to recognize. It must be simple and natural, and at the same time not primitive and not resembling other melodies.

Composition students are often given hints like: A good melody is long, has at least three types of durations, and contains both large and small intervals. Even such a general hint is not really true. The opening motive of Mozart's *Eine kleine Nachtmusik* is short and contains only large intervals. The choir melody from Beethoven's 9th Symphony has only quarter notes. The remarkable theme of Bach's E minor fugue from Book I of *Das wohltemperierte Klavier* is only two measures long and contains exclusively sixteenth notes (Figure 1 top).

Several publications on modeling melodies by a number of rules date back to the origins of cybernetics but the enthusiasm was soon exhausted; see Zaripov (1971, 1983) for review and original results. The lack of knowledge about what makes melodies good explains why the contemporary formal music prefers to avoid them at all. Since there are no criteria for their evaluation, there is no rule to sort out bad ones, and no algorithm to construct good ones.

My current work in progress attempts to partially fill in this gap with a model for evaluating melodies. The given paper outlines the idea of the approach developed.

While analyzing melodies, one is confronted with rhythm, pitch, and eventually with underlying harmony. Among these three factors, rhythm seems to be most important. Rimski-Korsakov (1844–1908) often repeated: "Rhythm is already music". He did not say anything similar about pitch or harmony, although he was a famous colorist.

However only the apparent rhythm of a melody, that is, the sequence of durations, is insufficient to make any serious judgement about the melody quality. In fact, a melody can have a poor rhythm but still be good (the choir from Beethoven's 9th Symphony, Bach's theme cited).

The *latent polyrhythm structure* of a melody, or simply its *polyrhythm*, seems to be more adequate to the task. To reveal it, the melody is regarded as polyphonic, that is, as constituted by *latent voices*; see Figure 1 bottom. Latent voices arise due to leaps, so that a melody with disjunct voice-leading is perceived as several voices with conjunct voice-leading. Since each latent voice has its own rhythm, we get an interaction of *latent rhythm layers*, constituting the polyrhythm of the polyphonic melody.

What is actually being done is similar to

Figure 2a

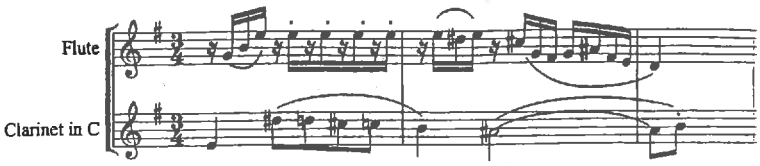


Figure 1: The theme of J.-S. Bach's E minor fugue from Book I of *Das wohltemperierte Klavier* and its latent voices

arranging the rhythm of a melody for a drum set with each drum playing a particular rhythm layer. Since the polyrhythm is recognized by melody leaps, certain pitch information is also taken into consideration. The harmonic factor is beyond the scope of the model.

The second step of analysis (most developed) is aimed at estimating the complexity of rhythm layers and of the total complexity of the polyrhythm as the sum of the layer's complexities. Below the corresponding model is described in more detail.

The polyrhythm of a good melody is assumed to have some 'optimal' complexity, neither too high, nor too low. It reflects the fact that a good melody must be enough simple but not primitive. Finding the lower and upper thresholds for the complexity indices of good melodies is a subject of psychological experiments which are not yet performed.

2 Pitch and polyrhythm

To explain why our melody analysis with no reference to pitch and harmony nevertheless makes sense, let us outline the evolution of musical pitch, also with regard to harmony.

Recall that early music traditions are mainly monophonic and pre-modal, that is, with no definite scale (Alekseev 1976). The pitch served for enhancing the intonation, e.g. in singing declamation, or for more dramatic effects like register changes. Such imprecise *intonation pitch* and *register pitch* are inherent in all early music traditions (Alekseev 1986).

The *modal pitch* with several degrees is typical for more advanced musical cultures. In early Western music, modes were used in a rather free way and could replace one another. Since music was still monophonic and tones were not assumed to sound simultaneously, the accuracy of pitch was not very critical.

Harmony has been 'invented' in Europe in the 15–16th centuries together with innovations in scales and tuning systems facilitating polyphony (Honneger 1996, p. 448). Harmony is often opposed to melody because it adds a vertical aspect to horizontal development. However, it is not that simple. Harmony has horizontal effects because successive and even distant tones create harmonic sensations. Melody against harmony was the point of a famous discussion between Rousseau and Rameau (Holopov 1976, p. 523). Rousseau subordinated harmony to melody, having compared the melody with the drawing in painting and the harmony with the color. Rameau argued that harmony, like a road, indicates the way and thereby gives birth to melody.

Which consequences had the invention of harmony for melodic thinking? Owing to certain harmonic advantages, major and minor scales gained superiority over other modes. Melodies became tonal (as opposed to modal and atonal) with particular gravity tones and a specific function of every scale degree. The pitch classes became fixed and at the same time fine tuned by backing harmonies, e.g. C in chord $D7$ is intonated lower than in chord C.

Harmonic deviations gave a means to control tonal tension thus enabling constructing long melodies. Finally, harmony makes a melody most beautiful and thus becomes its attribute (monophonic music with no harmony is less concerned with the 'beauty' of melody but rather with mood, intonation, contrasts, timbre, rhythm, and certainly words). Therefore, harmonic pitch with underlying harmonic relations is the most comprehensible form of musical pitch.

One can see that the register and intonation pitch have successively evolved into *modal*, *tonal*, and *harmonic pitch*, corresponding to the context where the pitch has been considered. The pitch, having become more accurate, was charged with more functional relations.

The historical perspective proves that *the register pitch is the primary form of pitch and, consequently, must be taken into account in the first turn.* This is exactly what is done while evaluating melodies by polyrhythms. Since the polyrhythm is revealed by leaps in the melody, the discrimination between large and small intervals is also implemented in the model.

Recognizing polyrhythm with respect to register pitch closely relates to perceptual streaming (Bregman and McAdams 1979, Bregamn 1990). Unlike perceptual classification (e.g. pitch recognition), streaming operates with permanently updated separating thresholds. For instance, the progression of tones c_1, c_2, d_1, $d_2, \ldots, c_3$, c_4 is streamed into two voices, low $c_1, d_1, \ldots, c_3$ and high $c_2, d_2, \ldots, c_4$. Note that the low voice remains low voice even at c_3 which is much higher than the first note c_2 of the high voice. Streaming uses only grouping and simplicity principles and does not require any learning. Therefore, it relates to 'naive' perception rather than to knowledge-based 'intelligent' perception.

Figure 2 top displays a monophonic melody, containing exclusively eighth durations. Nevertheless it is rhythmically perceived in a more complex way. The effect of rhythmic variety emerges due to perceptual streaming of tones into latent voices with their own rhythms. The streaming is explicitly shown in R. Brunetti's score (Figure 2 bottom) which emphasizes the rhythm complexity of the polyphonic melody. Tapping the Brunetti's rhythms on three drums resembles the original theme, proving that certain melodic information is retained. Similarly, a melody remains recognizable even if it is hummed with inaccurate intonation.

Thus, evaluating melodies by reducing them to polyrhythms deals with most fundamental melodic features, rhythm (as recognized by Rimski-Korsakov) and primary register pitch (as discovered by Alekseev). On the other hand, it is based on 'naive' perceptual streaming. It makes our incomplete analysis consistent with perception priorities, as if we simulate the reaction of a musically inexperienced listener.

3 Formal analysis of rhythm

Applying the ideas of Kolmogorow (1965) to rhythm, define the complexity of a rhythm to be the amount of data storage required for the algorithm of the rhythm generation (Tangian 1993, 1994, 1998). For instance, a repetitive rhythm requires storing its generative pattern and the number of repeats. If the rhythm of a melody uses several patterns, all of them must be stored, implying an increase in the rhythm complexity. In certain cases one rhythm pattern can be derived from another, which reduces the total complexity.

Generative rhythm patterns are not necessarily linked to measures and bar lines. Therefore, revealing them needs a special techniques which is implemented in the rules described below.

3.1 Accentuation

Rule 1 (Durations) *The only characteristic of a time event is the duration of time interval between its onset and the onset of the next time event. This inter-onset time interval is said to be the duration associated with the event. If the given time event is the last in the sequence, the associated duration is assumed to be not fixed.*

Rule 2 (Accentuation Distinguishability) *In order to distinguish accentuated events*

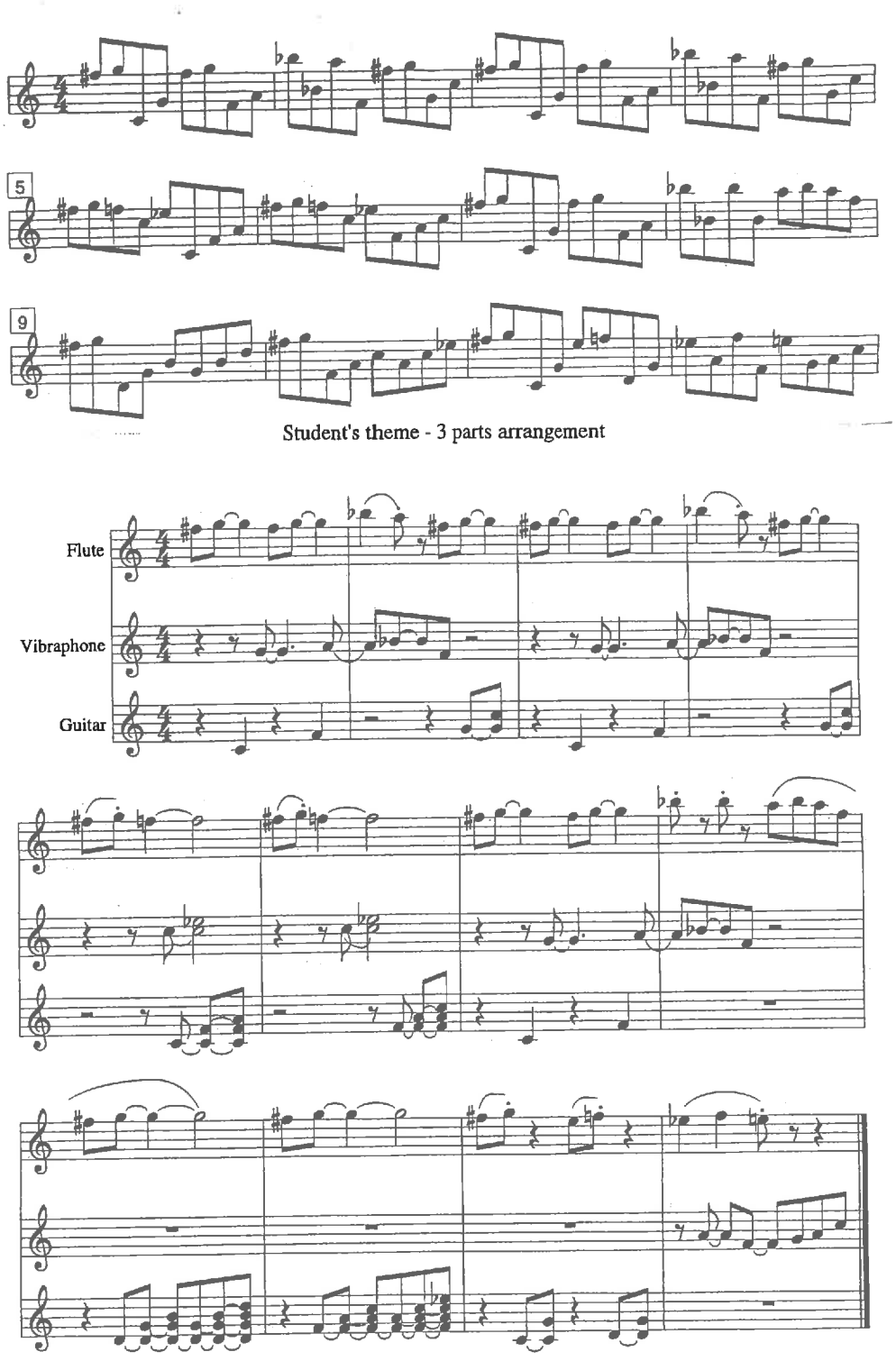


Figure 2: The *Student's Theme* by A. Tangian and its polyphonic arrangement by R. Brunetti's

in a sequence of time events, at least two types of durations are necessary.

Rule 3 (Accentuated Durations) *The duration associated with an event is said to be strongly accentuated if*

(a) it is longer than its closest neighbors, that is, it follows a shorter duration and the next one is also shorter;

(b) it follows an equal duration and the next one is shorter.

The duration is said to be weakly accentuated if

(c) it follows a shorter duration and precedes an equal duration which is not strongly accentuated, that is, the second next is not shorter.

A time event is said to be accentuated (strongly or weakly) if the duration associated with the event is accentuated (strongly or weakly, respectively).

The idea of the third rule is that a longer duration next to a shorter one is accentuated. The most evident Item (*a*) concerns a situation when a longer duration is between two shorter ones. If a duration is between shorter and equal one then it is usually accentuated (Items *b* and *c*), yet in order to avoid simultaneous accents at two equal successive durations between two shorter ones, we assume no accent at the first duration (Item *c*). No accentuation emerges when durations are successively getting shorter or longer.

In order to provide unambiguous segmentation when several accentuated durations emerge in a short phrase, we distinguish between strong and weak accents, with the priority of strong accents. We suppose that a change from a longer duration to a shorter one is immediately recognized, resulting in an accent. Yet after a change from a shorter duration to a longer duration one can expect some further increase in duration, resulting in a weaker sensation of accent; this is the case of weak accentuation.

Figure 3: Accentuation by timing cues

To illustrate the above rules, consider the sequence of time events shown in Figure 3. The first crotchet marked by symbol ">" is weakly accentuated by virtue of Rule 3c, since it is the duration between a shorter duration and an equal one. The last crotchet which is also marked by ">" is strongly accentuated by virtue of Rule 3b, since it is the duration between an equal and a shorter one. Note that although the second crotchet precedes an eighth, it is not accentuated. Indeed, by virtue of Rule 1 the duration of the event associated with this eighth is crotchet, and no shorter duration is adjacent to it. By virtue of Rule 1 the last note of the sequence can be considered both as accentuated, or not.

To show the accentuation in notation, bar lines are put before accentuated events as shown in Figure 3. In the above example, the segmentation with respect to the accentuation determines the 3/4 time of the given phrase.

3.2 Rhythm segmentation

Accentuation as defined above is not sufficient for rhythm segmentation. Indeed, consider a periodic sequence of time events segmented with respect to the accentuation defined in two different ways as shown in Figures 4a–b. To prove the perceptual ambiguity of segmentation of these events, we have performed the following audio experiment: The given sequence of time events has been recorded and played back in a loop, having been amplified gradually from zero level. A series of audio tests has shown that listeners recognize the two segmentations with almost equal probability.

Thus to recognize a rhythm segmentation we need some other cues in addition to the accentuation. For that purpose we introduce rules of classification and elaboration of rhythm patterns. By a *rhythm pattern* we understand any segment of a given sequence of durations.

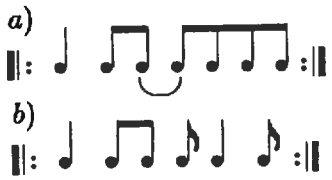


Figure 4: Rhythm segmentation by timing cues

Figure 5: Syllable as an indecomposable unit

However, the most important is the case when rhythm patterns are segmented with respect to the accentuation. Thus we obtain the following definition.

Rule 4 (Phrases and Syllables) *A rhythmic phrase is defined to be a sequence of durations which follows an accentuated duration and ends at an accentuated duration. A rhythmic phrase with the only accentuated duration is said to be a rhythmic syllable* (Katuar, 1926).

Consequently, a rhythmic syllable is a simplest rhythmic phrase. Any rhythmic phrase is formed by adding syllables to each other.

Note that by virtue of Rule 4 a syllable is determined by the durations which precede an accentuated event. The accentuated duration itself is not included into the syllable. The accentuated event just marks the end of the syllable, and the associated accentuated duration may be not fixed (cf. with Rule 1).

We suppose that rhythmic syllables are perceived as indecomposable time units. In order to prove it we have performed the following audio experiment: The rhythmic syllable shown in Figure 5 with two fixed absolute durations $0.1sec$ has been repeatedly reproduced under variable delays divisible by $0.1sec$, e.g. 0.8, 1.0, 1.2, 0.8, ... *sec.* If the rhythmic syllable was perceived as a composed structure, the common time unit ($0.1sec$) would result in a sensation of constant tempo with changes of rest durations. (Tempo determination with respect to the common time unit was proposed by Messiaen (1944)). However, in our audio experiment listeners have recognized tempo deviations rather than rhythm changes. This proves that the syllable is perceived as an entirety rather than as composed of smaller units and that the end duration is not important for identifying equal syllables.

Such a way of recognizing tempo by time intervals between the entries of similar rhythm patterns meets the principle of correlativity of perception. In fact, in our experiment we have shown that similar rhythm patterns are used as reference indivisible units for tempo tracking. Besides, we have shown that the tempo is a percept of another level than the rhythm.

3.3 Operations on rhythm patterns

In order to classify rhythmic phrases and recognize generative rhythm patterns, we define a reflexive transitive binary relation E, *"is the elaboration of"* on the set of rhythm patterns X. Recall that a binary relation E on X is reflexive if xEx for all $x \in X$ (a rhythm pattern is the elaboration of itself), and transitive if xEy and yEz implies xEz for all $x, y, z \in X$ (a successive elaboration of a rhythm pattern is its elaboration).

Rule 5 (Elaboration) *Rhythm pattern A is the elaboration of rhythm pattern B if A preserves the pulse train of B, that is, if A results from a subdivision of durations of B by inserting additional time events* (Mont-Reynaud & Goldstein, 1985).

Figure 6 illustrates the idea of rhythm elaboration with an example of subdivisions of a crotchet duration (recall that by virtue of Rule 1 the crotchet duration, in order to be determined, should be followed by a next tone onset which is not shown in the figure).

The idea of elaboration can be explained in correlation terms. Represent the rhythm patterns in Figure 6 by 0 and 1 within the accuracy of a sixteenth. Then the top pattern which we

denote by T is written down as follows

$$T = \{t_1 \ldots t_4\} = \{1000\} \ ,$$

and the bottom pattern which we denote by B is written down as

$$B = \{b_1 \ldots b_4\} = \{1111\} \ .$$

It is easy to see that pattern B is the elaboration of pattern T if and only if $B \supset T$. This means that B contains all ones of T. Since the number of ones in T is equal to the autocorrelation

$$R_{T,T} = \sum_{i=1}^{4} t_i \cdot t_i$$

(which in the given case is equal to 1), and the number of coinciding ones in B and T equals to the correlation

$$R_{B,T} = \sum_{i=1}^{4} b_i \cdot t_i$$

(which in the given case is equal to 1), we obtain that B is the elaboration of T if and only if

$$R_{B,T} = R_{T,T} \ .$$

Since the correlation is usually understood as a measure of similarity, the last equation means that the pattern B, being the elaboration of pattern T, is similar to pattern T.

For the patterns of equal duration which are not the elaboration of each other (as in the second line of Figure 6), the correlation is less than autocorrelation. For example, putting

$$L = \{l_1 \ldots l_4\} = \{1100\}$$

and

$$M = \{m_1 \ldots m_4\} = \{1010\} \ ,$$

we obtain

$$\begin{aligned} R_{L,M} &= \sum_{i=1}^{4} l_i \cdot m_i = 1 \\ &< 2 \\ &= R_{L,L} = \sum_{i=1}^{4} l_i \cdot l_i \\ &= R_{M,M} = \sum_{i=1}^{4} m_i \cdot m_i \ . \end{aligned}$$

Now we define the junction of syllables.

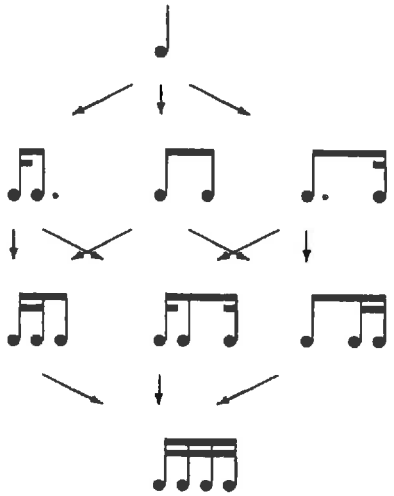


Figure 6: The elaboration of a crotchet rhythm pattern

Rule 6 (Sum and Junction of Syllables) *The sum of two successive rhythm patterns is defined to be the rhythm pattern constituted by the time events of these patterns which are put one after another.*

The junction of two successive rhythmic syllables is defined to be a rhythmic syllable which is the elaboration of their sum.

Note that the sum of two syllables is more than the two syllables in succession. Besides the two syllables themselves, the sum contains the *link*—the accentuated duration after the first syllable. According to the remark following Rule 4, this duration is undefined if the first syllable is considered separately, since instead of the whole duration we consider just an accent. In the sum of syllables, this accent turns to be a duration, linking the two syllables. Therefore, there can be many different sums of the same two syllables, depending on the link duration.

Also note that the sum of two rhythmic syllables is a rhythmic phrase, whereas their junction is a rhythmic syllable. This means that the sum of two syllables can have two accents, at the ends of each syllable, whereas in their junction the internal accent is suppressed by dividing the associated duration into shorter ones

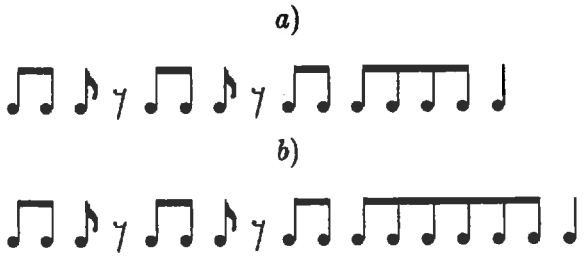


Figure 7: Two different junctions of the same rhythmic syllables

which are no longer accentuated. This implies that the junction has a new rhythm quality, a through tension towards its end.

Figure 7a displays two identical rhythmic syllables and their junction. The total duration of the third syllable is the same as the sum of the two syllables. This results in the symmetry of the whole passage, providing its structure to be $1 + 1 + 2$.

Consider another junction of the two syllables, for instance, obtained by adding two quavers to the third syllable as shown in Figure 7b. This implies that we link the first two syllables not by a crotchet duration but by a half-note duration, that is, we consider the elaboration of another sum of the syllables.

The connection between the three syllables in Figure 7b is less evident than in Figure 7a. Indeed, in Figure 7a one can see not only the two syllables, but already their sum which is elaborated next. In a sense, the elaboration is already "prepared" for easy perception. On the contrary, in Figure 7b the sum of the two syllables is different from the sum which is elaborated. In Figure 7b the intermediate phase between the two syllables and their junction is missed, breaking the successiveness in their perception.

We could provide the effect of such a successiveness in Figure 7b, making the rest between the first two syllables longer, up to a half-note duration. The duration of the second rest is not so important. Even if we change the duration of the second rest in Figure 7a, the third rhythmic syllable is still perceived as the elaboration of the sum of the first two.

From our standpoint, we can explain the simplicity of rhythm construction $1 + 1 + 2 + 4 + \ldots$. Such a structure contains a rhythm pattern, the elaboration of the preceding segment, then the elaboration of two preceding segments, and so on. Therefore, the origin of such a structure is quite simple, adding junctions of all preceding segments. This results in perceiving such rhythms with ease; moreover, the perception is "prepared" to recognize the elaboration since the sum is already exhibited.

3.4 Time and rhythm complexity

Thus we have introduced the rules of representation of a given sequence of time events in terms of generative syllables. Constructing such representations, one can reveal origins of a given rhythm with conclusions concerning its time.

Note that rhythm patterns of equal total duration constitute an *ordered directed set* with respect to the elaboration, where every two elements have a common superior—their common *root*. An example of such an order is shown in Figure 6 with a common root pattern at the top and its successive elaborations indicated by arrows.

The patterns of the same total duration which are not elaborations of each other (like in the second line of Figure 6) are of particular interest. If a rhythm contains such patterns then this rhythm has no embedded levels of the pulse train and can be represented as a succession of irreducible units whose pulse train becomes predominant.

The idea of a pulse train generated by indecomposable rhythm patterns can be applied to rhythmic syllables. Since each syllable has the only accent, the accents of syllables determine a pulse train with a certain rhythm. We use this rhythm to determine the time of a given sequence of time events.

Rule 7 (Determination of Time) *If a sequence of time events is representable in terms of elaboration of certain rhythmic syllables (phrases), then the time of the given sequence is determined by the duration ratio of their roots.*

In other words, one has to find a stable preimage (with respect to the elaboration) of generative patterns.

Roughly speaking, the time is defined to be the rhythm of roots of generative syllables.

Besides time determination, rhythmic patterns which are irreducible to each other can be used for estimating the complexity of rhythm. Indeed, their number corresponds to the number of generative patterns required to generate the given sequence.

For example, consider the rhythm in Figure 8 which is constituted by two rhythm patterns of equal duration. One can see that the crotchet duration is the root for the two rhythm groups beamed but no rhythm group is the elaboration of another. This means that the pulse train of crotchets is supported by no pulse train of quavers or some other shorter durations.

Such a rhythm can be considered as less redundant and therefore as more complex. The *complexity of a rhythm* can be identified with the branching index of the graph of the rhythm patterns used, that is, by the maximal number of irreducible to each other rhythm configurations of the same level. For instance, the rhythm in Figure 8 is generated by two patterns of equal duration which are not reducible to each other; consequently, its complexity index is equal to 2.

Such an understanding of rhythm complexity meets the ideas of Messiaen (1944) who has characterized the variety of rhythm by the number of non-commensurable patterns used.

Thus finding irreducible (with respect to elaboration) patterns has two applications: time recognition and estimation of rhythm complexity.

3.5 Example of estimation

Consider the snare drum part from *Bolero* by M. Ravel (Figure 9). Since we use time data only (Rule 1), our method cannot be applied to a rhythm which is based on pitch and dynamic accentuation. Since the chosen rhythm contains two types of durations, by virtue of

Figure 8: A rhythm with complexity 2

Rule 2 it is an appropriate object for our analysis. Let us trace the procedure of structurizing this rhythm step by step.

1. Consider Duration 0. The following one is shorter, consequently, by virtue of Rule 3b it is strongly accentuated. Since it is the first event in the sequence, we recognize the first syllable S as constituted by Duration 0 only. To write down the syllables, we shall use the denotations from Section 3.3, with the only difference that a digit will correspond not to the duration of sixteenth but to the duration of sixteenth triplet. Thus,

   $S = \{100\}$, that is, .

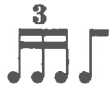
   Thus up to the current moment our rhythm is represented by the only syllable

   S .

2. Consider Duration 1. It is preceded by a longer duration and succeeded by an equal one. By Rule 3 it is not accentuated. By Rule 4 we don't recognize the end of a syllable at Duration 1.

   Since Durations 2 and 3 are not preceded or succeeded by shorter ones, by virtue of Rule 3 they are not accentuated. Since they are not accentuated, by Rule 4 we don't recognize the end of syllable at these durations.

3. Since Duration 4 is between two shorter durations, by virtue of Rule 3a it is strongly accentuated. By Rule 4 we recognize the end of syllable which we denote

   $S_1 = \{111\ 100\}$, that is,  .

   Now we compare syllable S_1 with the earlier recognized, verifying:

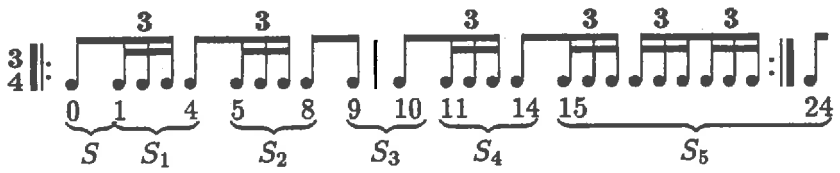


Figure 9: Determination of time by recognizing rhythmic syllables

(a) whether the given syllable is the elaboration of another one;

(b) whether any other syllable is the elaboration of the given one;

(c) whether the given syllable is the junction of other syllables;

(d) whether any other syllable is the junction of the given syllable with another one.

One can see that syllable S_1 is not the elaboration of any other syllable, no syllable is the elaboration of S_1, but S_1 is the junction of two syllables S. Therefore, up to the current moment our rhythm is represented as

$$S \;\; S_1 \;\; ,$$

or

$$S \;\; E(2S) \;\; ,$$

where $E(2S) = E(S+S)$ denotes the elaboration of the sum $S+S$ (that is, the junction of two syllables S).

4. Similarly to Item 2, there is no accentuation at Durations 5–7 and we don't recognize the end of syllable.

5. Similarly to Duration 4 analyzed in Item 3, there is an accentuation at Duration 8, with the only difference that Duration 8 is *weakly* accentuated. By Rule 4 we recognize the end of syllable which we denote

$S_2 = \{111\ 100\}$, that is, 

Note that syllable S_2 is equal to S_1. Consequently, everything said about syllable S_1 relates also to S_2. Therefore, up to the current moment our rhythm can be represented in the following two ways

$$\begin{array}{llll} S & E(2S) & E(2S) & ; \\ S & S_1 & S_1 & . \end{array}$$

6. One can see that Duration 9 is not accentuated, and therefore no syllable ends at Duration 9.

7. By Rule 3b Duration 10 is accentuated, and we recognize syllable

$S_3 = \{100\ 100\}$, that is, 

Answering the questions (a)–(d) enumerated in Item 3, we recognize that S_1 and S_2 are the elaborations of S_3; besides, S_3 is the sum of two syllables S. Thus we obtain the following equivalent representations of our rhythm:

$$\begin{array}{llllll} S & E(2S) & E(2S) & 2S & ; \\ S & E(S_3) & E(S_3) & S_3 & . \end{array}$$

8. Since Durations 11–13 are not accentuated, no syllable ends at these durations.

9. Since by Rule 3a Duration 14 is accentuated, we recognize syllable

$S_4 = \{111\ 100\}$, that is, 

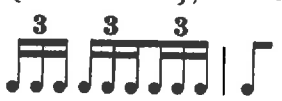
Having answered the questions (a)–(d) enumerated in Item 3, we obtain the following representations of the rhythm:

$$\begin{array}{lllll} S & E(2S) & E(2S) & 2S & E(2S) \;; \\ S & E(S_3) & E(S_3) & S_3 & E(S_3) \;. \end{array}$$

10. Since Durations 15–23 are not accentuated, no syllable ends at these durations.

11. By virtue of Rule 3a Duration 24 (or Duration 0, taking into account the repeat sign) is accentuated. Consequently, we recognize syllable

S_5 = {111 111 111}, that is,



Having answered the questions (a)–(d) enumerated in Item 3, we obtain that

$$S_5 = E(S_1{+}S_3) = E(S_2{+}S_3) = E(S_3{+}S_3).$$

Hence, we get the following two representations of our rhythm:

$$S\,\|\!:\, E(2S)\; E(2S)\; 2S\; E(2S)\; E(4S)\; :\!\|\;;$$

$$S\,\|\!:\, E(S_3)\; E(S_3)\; S_3\; E(S_3)\; E(2S_3)\; :\!\|\;,$$

or

$$S\,\|\!:\, S_1\; S_1\; S_3\; S_1\; E(S_1+S_3)\; :\!\|\;. \quad (1)$$

If we consider strong accents only, ignoring weak accents, then syllables S_2 and S_3 join into syllable

S_{2+3} = {111 100 100 100}, that is,

Since rhythmic syllable S_5 is the junction of syllables S_2 and S_3, we obtain even more simple representation of the rhythm as follows

$$S\;\|\!:\; S_1\;\; S_{2+3}\;\; S_1\;\; E(S_{2+3})\;\; :\!\|\;. \quad (2)$$

With regard to the repetitions of the given rhythm, syllable S can be interpreted as the end of syllable S_5. Finally, we obtain the representation of the given rhythm as generated by phrase S_1, S_{2+3}. Since S_{2+3} is two times longer than S_1, by virtue of Rule 7 we interpret our rhythm as having triple time: 3/4, or 3/8, etc. The choice of denominator (unit of counting) is a question of convention.

Note that there is a risk to interpret the period in (1) as consisting of three equal groups, that is, instead of "correct" segmentation

$$S\,\|\!:\, [S_1\; S_1\; S_3]\; [S_1\; E(S_1+S_3)]\; :\!\|\;,$$

Table 1: Complexity of the *Bolero* rhythm

Operation	Complexity
Elaboration of S_3 by 2 durations	1+2
Repeat of $E(S_3)$	1
Coding of S_3	2
Repeat of $E(S_3)$	1
Sum of $E(S_3)$ and S_3	1
Elaboration of $E(S_3)+S_3$ by 2 durations	1+2
Total complexity	11

on can accept the "wrong" segmentation

$$S\,\|\!:\, [S_1\; S_1]\; [S_3\; S_1]\; [E(S_1+S_3)]\; :\!\|\;.$$

This corresponds to recognizing the time of the rhythm as 2/4. However, the representation (2) which is obtained by ignoring local accents leaves no doubts in the triple time basis. Thus distinguishing between strong and weak accents is rather useful.

Since S_1 is elaboration of S_3, we obtain

$$S\,\|\!:\, [E(S_3)\; E(S_3)\; S_3]\; [E(S_3)\; E(E(S_3)+S_3)]\; :\!\|$$

Assume that the complexity $C\{S_3\}$ is two bytes (= two durations), and that calling the algorithms of repeat, of sum, and of elaboration require 1 byte each, we obtain the complexity of the rhythm inside the repeat signs as shown in Table 1.

4 Summary

1. It is supposed that a polyphonic melody can be evaluated by estimating the complexity of its polyrhythm.
2. A basic model for estimating the complexity of (poly)rhythm is proposed.
3. Psychological experiments on establishing upper and lower complexity thresholds characterizing good melodies are planned together with improvements the model.
4. The method is intended for several applications, including computer composition.

References

ALEKSEEV E.Y. (1976): *Problems of the mode's formation*, Sovetskii Kompozitor, Moscow (Russian).

ALEKSEEV E.Y. (1986): *Intonating in early traditional music: the pitch aspect*, Sovetskii Kompozitor, Moscow (Russian).

BREGMAN A.S. (1990): *Auditory Scene Analysis: The Perceptual Organization of Sound.* Cambridge, Massachusetts: M.I.T. Press.

HOLOPOV Y.N. (1976): "Melody", *Music Encyclopedia, Vol. 3*, Soviet Encyclopedia, Moscow, 511–529 (Russian).

HONNEGER M. (ED.) (1996): *Connaissance de la musique,* Bordas, Paris. (1st Ed. 1966.)

KATUAR G. (1926): *Muzykalnaya Forma. 1. Ritm.* Moscow. (Russian).

KOLMOGOROV A.N. (1965): Three Approaches to Defining the Notion "Quantity of Information". *Problemy Peredatchi Informatsii,* 1(1), 3–11. Reprinted in: Kolmogorov A.N. *Theory of Information and Theory of Algorithms.* Moscow: Nauka, 1987, 213–223. (Russian).

LERDAHL F. & JACKENDOFF R. (1983): *A Generative Theory of Tonal Music.* Cambridge, Massachusetts: M.I.T. Press.

MCADAMS S., & BREGMAN A. (1979): Hearing musical streams. *Computer Music Journal,* 3(4), 26–43, 60, 63. Peprinted in: C. ROADS & J. STRAWN (EDS.) (1985) *Foundations of computer music*, MIT Press, Cambridge Mass., 658–698.

MESSIAEN O. (1944): *Technique de mon langage musical. Vol. 1.* Paris: Leduc.

MONT-REYNAUD B. & GOLDSTEIN M. (1985) On Finding Rhythmic Patterns in Musical Lines. *Proceedings of the International Computer Music Conference'1985.* San Francisco: Computer Music Association, 391–397.

TANGUIANE A.S. (1993): *Artificial Perception and Music Recognition.* Berlin: Springer-Verlag (Lecture Notes in Artificial Intelligence No. 746).

TANGUIANE A.S. (1994): A Principle of Correlativity of Perception and Its Applications to Music Recognition. *Music Perception,* 11 (4), 465–502.

TANGIAN A.S. (1998): An operational definition of rhythm and tempo. *General Psychology,* Vol. ** (Temporal Dynamics and Cognitive Processes), 55–89.

ZARIPOV R.H. (1971): *Cybernetics and music,* Nauka, Moscow (Russian).

ZARIPOV R.H. (1983): *Computer generation of variants for modeling human creativity,* Nauka, Moscow (Russian).

Un modèle d'analyse pour les musiques électroacoustiques

Pierre Couprie

Résumé : Cet article décrit un modèle d'analyse élaboré par l'auteur. Ce modèle est basé sur la description détaillée des sons et des structures sonores et/ou musicales afin de faire apparaître les relations complexes qui sous-tendent une œuvre électroacoustique. Plusieurs logiciels sont employés dans ce travail. Tout d'abord Conversion (communication entre l'Acousmographe et d'autres logiciels) et SONalyse (base de donnée pour la description des sons) développés par l'auteur et ensuite l'Acousmographe (Ina-GRM) et Open Music (IRCAM) pour la représentation et l'analyse des données.

Mots clés : analyse musicale, électroacoustique, segmentation, critères, Acousmographe, Open Music, analyse de données

I. Introduction

Je consacre mes recherches depuis trois ans à l'analyse des musiques électroacoustiques et plus particulièrement ce que l'on nomme parfois les musiques de support ou plus couramment la musique concrète. De support, car, pour le son, il n'y en a qu'un : la bande magnétique ou le fichier numérique. Il n'y a pas de partition : ce qui serait inutile vu qu'il n'y a pas d'instrumentiste ; tout au plus quelques relevés textuels ou graphiques réalisés par le compositeur. C'est la nature même de cette musique qui soulève, à la base, quelques problèmes pour le chercheur. François Delalande nous fait remarquer :

> " [...] cette musique pose à l'analyste tous les problèmes à la fois : pas de partition, pas de système, pas d'unités "prédécoupées" comme les notes. Nous sommes devant le "cas général" dont les autres musiques (musiques écrites musiques traditionnelles) apparaissent comme des cas particuliers simplifiés."[1]

Malgré cette multiplication des problèmes, plusieurs chercheurs ou groupes de chercheurs se sont penchés sur l'analyse de la musique concrète. Certains ont même été jusqu'à élaborer de véritables outils de description du sonore et/ou du musical :

> 1) le premier de ceux-ci : Pierre Schaeffer a développé les notions d'*objets sonores* et d'*écoutes réduites* afin de guider le travail du compositeur. Il dote l'analyse sonore d'un puissant outil de description morphologique ;
> 2) Murray Schafer a exploré au Canada la description de *paysage sonore* en mettant en avant la notion de *fait sonore*. Chez lui, le son n'est pas coupé de sa référence, contrairement à l'objet sonore de Pierre Schaeffer ;
> 3) Stéphane Roy, autre Canadien, a cherché, quant à lui, à verbaliser les fonctions d'un certain nombre d'objets sonores. Ses analyses se présentent comme une

[1] Delalande, François, *Analyse musicale et conduites de réception : "Sommeil" de Pierre Henry*, inédit, p.3

collection "d'objets-balises" mettant de mettre l'accent sur des moments cruciaux de l'évolution structurelle de l'œuvre ;
4) Le Néerlandais Lasse Thoresen a exploré les articulations sonores avec ses *champs temporels* ;
5) enfin, le groupe du MIM, sous la direction de Marcel Frémiot, a tenté de mettre en place un système de description fin de révéler les dimensions temporelles et sémiologiques des sons : les *Unités Sémiologiques Temporelles*.

L'absence de partition a certainement été un frein au développement de l'analyse des musiques électroacoustiques car elle nous oblige à élaborer une recherche exclusivement basée sur la perception. Toutefois, cette absence de support visuel n'est pas tout à fait réelle. En effet, le chercheur utilise aujourd'hui très souvent le sonagramme comme une analyse acoustique de base permettant, à l'aide de diverses annotations, de faire un premier repérage temporel des différents éléments de l'œuvre (voir le graphique 1 en annexe).

Les facilités d'accès à des outils informatiques permettant de construire une représentation physique (le sonagramme) ou musicale (l'Acousmographe[2] et les logiciels multimédias) de l'analyse devraient la rendre plus abordable dans les années à venir.

Sur ce graphique, nous observons deux plans : l'un, en haut, étant le sonagramme, c'est-à-dire la représentation des fréquences en trois dimensions (la hauteur sur l'axe des ordonnés, les temps sur l'axe des abscisses et l'intensité en niveau de gris). L'autre, en bas, représente l'intensité globale des deux canaux (droite et gauche). Notons au passage qu'en électroacoustique, nous n'avons pratiquement jamais affaire à une véritable stéréophonie mais à ce que l'on pourrait nommer une "bi-phonie".

Le second problème relevé par François Delalande concerne la segmentation. Loin d'être impossible, elle demeure toutefois délicate à réaliser car elle reste basée sur un choix intuitif de la part du chercheur. Pierre Schaeffer s'était bien sûr déjà heurté au problème. Il l'avait résumé dans le couple objet/structure (voir le graphique 2 en annexe).

Chaque son peut être analysé comme faisant partie d'un ensemble plus vaste : une structure d'objet sonore. Mais il peut aussi être décomposé en des éléments plus fins, en critères typo-morphologiques[3] par exemple. Selon le grossissement de notre loupe auditive, le travail sera plutôt orienté vers une analyse fine des unités sonores ou vers une analyse globale des structures. Disons-le tout de suite : il n'y a pas de solution miracle. Chaque œuvre implique différentes options possibles selon la direction dans laquelle le chercheur désire s'orienter. La seule contrainte pour l'analyste est de ne pas changer, en cours d'analyse, de niveau d'observation. Il peut par contre multiplier ces niveaux afin de les mettre en relation.

---

[2] Voir une description de l'Acousmographe dans la partie III ou dans :
Couprie, Pierre, Battier, Marc, "L'*Acousmographe* : un outil pour l'analyse informatique de documents sonores", *Les cahiers de l'O.M.F.*, n°4, Paris, Université de Paris IV-Sorbonne, 2001, en cours de publication.
[3] Pierre Schaeffer a décrit 7 critères typo-morphologiques : la masse, la dynamique, le timbre harmonique, le profil mélodique, le profil de masse, le grain et l'allure.

II. l'extraction des critères en fonction de l'œuvre et de l'orientation analytique

II.1. La recherche de critères d'analyse

Le travail du chercheur désirant analyser une œuvre électroacoustique commence par l'extraction d'un certain nombre de signes musicaux signifiants[4]. Ces signes auront pour rôle de déterminer assez rapidement d'une part, quel sera le niveau d'analyse souhaitable et d'autre part quels seront les types d'unités sonores prises en compte. En fonction de ses intentions premières et des signes musicaux signifiants, le chercheur établira un ou plusieurs critères généraux permettant de choisir, dans le flux sonore, les éléments à analyser. Ils peuvent êtres classés en trois catégories :

1) les éléments sonores ou musicaux à analyser minutieusement ;
2) les éléments à analyser globalement et susceptible d'apporter des renseignements pour l'analyse des éléments de première catégorie;
3) enfin, les éléments à mettre côté ou ne pas analyser.

Les deuxièmes et troisième catégories peuvent êtres supprimées si le chercheur décide de produire une analyse aussi exhaustive que possible.

Dans la deuxième étape, l'analyste élabore une liste de critères morphologiques, référentiels ou structurels pouvant prendre un certain nombre de valeurs déterminés. Nous allons maintenant observer ces critères de plus près.

II.1.1. Les critères morphologiques

Ils correspondent à la description du sonore sur le plan de l'acoustique. Pierre Schaeffer[5] et Denis Smalley[6] ont été les seuls jusqu'à présent à les étudier. Les critères qu'ils ont révélés ne sont pas tous utilisables lors d'une analyse musicale, j'en ai donc retenu quelques-uns en les regroupant sous 4 catégories :

1) le spectre : type, registre, profil mélodique et profil de masse, allure ;
2) la dynamique : attaque, déclin, maintien, relâchement, écart, cycle ;
3) le grain : type ;
4) l'espace interne : place, espaces de départ et d'arrivée, mouvement.

II.1.2. Les critères référentiels

Je me suis inspiré de deux types de recherches : d'une part, celle réalisée par Murray Schafer[7] au Canada sur la description des paysages sonores et d'autre part celle élaborée afin de qualifier les voix[8] grâce à des différentiateurs sémantiques. Ainsi, le critère de référence contient plusieurs sous-catégories :

---

[4] Imberty, Michel, *Entendre la musique, sémantique psychologique de la musique*, Paris, Dunot, 1979, p.13.
[5] Schaeffer, Pierre, *Traité des objets musicaux*, Paris, Le Seuil, Pierres Vives, 3/1977, 712 pp.
[6] Smalley, Denis, "Spectro-morphology and Structuring Processes", *The Language of Electroacoustic Music*, Londre, Simon Emmerson/The Macmillan Press LTP, 1986, pp. 61-93.
[7] Schafer, Murray, *Le paysage sonore*, Paris, J.C.Lattès, 1979, 391 pp.
[8] Payri, Blas, *Perception de la voix parlée : cohérence du timbre du locuteur*, Thèse de Doctorat, Université de Paris-Sud, 2000, 313 pp.

1) structure : fond/figure, émergence, échange ;
2) type référentiel à travers des catégories générales (bruits de nature, bruits humains, etc.) ou précises ;
3) qualification de l'effet utilisé ;
4) présence d'un texte : brassage, rythme, type, nouveau texte ;
5) émotion.

II.1.3. les critères de structure

La recherche et l'élaboration de critères d'ordre structurels sont une activité ingrate. En effet, elle est souvent considérée comme extrêmement subjective et est, par conséquent, très critiquée ; et pourtant quoi de plus subjectif qu'une analyse musicale ? Trois recherches très intéressantes ont été menées dans ce domaine : Les UST[9] autour de Marcel Frémiot, les objets-fonctions[10] de Stéphane Roy et les *Champs temporels*[11] de Lasse Thoresen. Ajoutons aussi les réflexions de François Bayle sur les causalités physiques, perceptuelles ou sensibles[12] d'une structure sonore.

Plusieurs entrées sont possibles afin de regrouper ces différents éléments :

1) catégories en temps / hors temps ;
2) catégories fonction formelle / articulation structurelle ;
3) catégories causalité physique (concret) / causalité sensible (abstrait) et structures imaginaires.

Observons maintenant comment réaliser techniquement la collecte des valeurs que peuvent prendre ces différents critères.

II.2. L'élaboration d'une fiche de description modulaire des unités sonores

Chaque œuvre et chaque recherche nécessite une focalisation particulière sur certains éléments musicaux. Je suis donc en train d'élaborer une grille modulaire d'analyse musicale. Les critères sont répartis non seulement selon plusieurs catégories sonores ou musicales mais aussi selon le type de matériau analysé (présence ou non de voix, de texte, de citations musicales ou de références). Ainsi cette grille à double entrée permet de décrire avec plus ou moins de précision (selon l'intention du chercheur) les différents éléments sonores et/ou musicaux segmentés de l'œuvre.

Cette base de données[13] permet de guider l'utilisateur pour le choix des catégories et des critères à intégrer dans une fiche de description finale. Un ensemble de questions aide le

---

[9] (Collectif) *Les Unités Sémiotiques Temporelles, éléments nouveaux d'analyse musicale*, Marseille, MIM, 1996, 96 pp.
[10] Roy, Stéphane, "Analyse des œuvres acoustiques : quelques fondements et proposition d'une méthode", *Electroacoustique-Québec : l'essor, Circuit, Revue nord américaine de musique du XX° siècle*, vol. 4, n°1-2, Montréal, Les Presse de l'Université de Montréal, 1993, pp. 67-91.
[11] Thoressen, Lasse, "Auditive Analysis of Musical Structures. A summary of analytical terms, graphical signs and définitions", *ICEM Conference on Electro-acoustic Music*, Stockholm, Royal Swedish Academy of Music, 1985, pp.65-90.
[12] Bayle, François, propos du *Séminaire « L'infini du bruit »*, Ina-GRM, 2001.
[13] Elle se présente sous la forme d'un logiciel programmé en BASIC et en cours de développement : SONalyse.

chercheur dans son choix. A tout moment, ce dernier peut ajouter une catégorie ou un critère avec un ensemble de valeurs possibles. La sauvegarde se fait sous forme de base de données ou de fichier texte. Une fois la fiche déterminée, le logiciel peut enregistrer les différentes fiches correspondant aux sons analysés.

Voici deux exemples de type de fiche[14], l'une utilisée dans l'analyse d'une œuvre d'Alain Savouret (extrait de *Don Quichotte Corporation*) et l'autre, élaborer pour analyser les caractéristiques de la voix dans une pièce de Jacques Lejeune (*Le Cantique des Cantiques*). La première se veut relativement exhaustive, mais vous pourrez remarquer (voir le graphique 3 en annexe) qu'il manque certaines catégories de critères essentiels. En effet, le compositeur utilise des sons ne possédant pas par exemple d'allure[15]. dans la seconde fiche (voir le graphique 4 en annexe), l'orientation analytique était volontairement tournée vers la voix et sa relation au matériau environnant. C'est pourquoi de nombreuses caractéristiques morphologiques des sons sont volontairement mises de côté pour privilégier des critères de description de la qualité de la voix, de l'émotion qu'elle suggère et du travail du compositeur sur le texte.

III. L'annotation du fichier sonore et l'analyse des données

III.1. L'analyse des données

L'Acousmographe est un logiciel développé par l'Ina-GRM pour les plates-formes Mac et PC. Il permet de réaliser le sonagramme d'un fichier sonore (œuvre complète ou extrait d'œuvre) et de l'annoter sous forme de graphiques et/ou de textes. Dans l'exemple utilisé ici, j'ai réalisé sur un premier plan une représentation graphique du son et, sur un second plan une annotation textuelle représentant les différents critères choisis pour analyser la voix et ses interactions avec les autres sons dans *Le Cantique des Cantiques* de Jacques Lejeune. L'ensemble des données (graphique et texte) dessinées sur l'Acousmographe peut être récupéré afin d'être analysé grâce au logiciel Conversion[16].

Le texte correspondant à la description de la voix, récupéré grâce au logiciel Conversion, est transposé sous forme de base de donnée afin d'obtenir une représentation graphique de la répartition des différentes valeurs en fonction de la structure temporelle de l'œuvre. Ces représentations graphiques en 2 ou 3 dimensions sont réalisées sur Excel ce qui permet de mettre en relief certains éléments de l'œuvre noyés dans le flux sonore lors de l'écoute. Dans le graphique 5 (voir en annexe), nous observons la répartition des différents éléments vocaux du *chant n°3* du *Cantique des Cantiques* par le calcul des moyennes et variances sur les critères présentant des valeurs linéaires ou discrètes. Open Music, quant à lui, me permet d'automatiser certaines analyses de données afin de comparer des chaînes de critères (calculs de statistiques, de distances ou d'associations paradigmatiques).

III.2. La classification

[14] Les deux fiches présentées ici sont dans un format de tableur et non dans celui de SONalyse.
[15] Variation plus ou moins régulière et cyclique de la hauteur d'un son.
[16] Conversion est un petit logiciel que j'ai développé en BASIC permettant d convertir les fichiers texte et bibliothèque obtenus avec le plu gin "transfert" de l'Acousmographe. Le résultat peut être sauvegardé sous trois types de formats :
1) texte pour l'impression ;
2) listes pour Open Music ;
3) base de données pour un tableur.

Une autre activité de l'analyse musicale consiste en la classification des différents sons ou structures sonores segmentées. Qui dit classification, dit mise en place d'un certain nombre de catégories pouvant regrouper plusieurs types de sons ayant une ou plusieurs qualités en commun. Ce type d'approche s'apparente à l'analyse paradigmatique tel que l'a défini Jean-Jacques Nattiez[17].

Je pense toutefois qu'une analyse plus proche de la perception musicale reflète mieux les relations entre les différents éléments du continuum sonore. Ainsi comparer ces éléments en les classant selon les valeurs de leur critères purement musicaux semble un peu simpliste. Je reste persuadé que, lors de la perception, l'auditeur associe non pas un ensemble de qualités sonores mais une étiquette correspondant à un critère dominant (voire plus rarement un ensemble de deux critères intimement mêlés). Ce critère dominant dépend de l'environnement de l'élément sonore et de sa place dans le déroulement temporel de l'œuvre. Il s'agit d'une sorte de *synecdoque* musicale. d'autre part, les émotions véhiculées par la musique me semble très importantes dans la perception d'une structure musicale. Ainsi, certaines structures sonores seront associées car elles évoquent le même images[18].

IV. Conclusion

Je viens de vous présenter une méthode d'analyse que j'ai élaborée pour les musiques électroacoustiques. Les résultats obtenus grâce à des techniques de statistiques ou d'observations fines de l'évolution de différents éléments du continuum sonore sont complémentaires d'une analyse plus "traditionnelle". En effet, certains éléments sont identiques à ceux trouvés lors d'une analyse purement auditive. Mais cette méthode apporte aussi des renseignements inédits sur la structure formelle ou sur l'évolution des morphologies des unités sonores.

Il y a un autre plan de l'analyse des musiques électroacoustiques dont je n'ai pas encore parlé et qui semble incontournable. Il s'agit de la présentation du travail analytique. Les supports multimédias qui se développent très rapidement et sont désormais facilement abordables pour le musicologue vont probablement, comme je le faisais remarquer dans l'introduction, donner un second souffle à l'étude de cette musique et à l'analyse musicologique en général. Le CD-Rom réalisé par l'Ina-GRM[19] fait découvrir, grâce encore une fois au support visuel, au néophyte ou au musicologue confirmé des plans d'écoutes, et donc d'analyse, qu'il était impossible de présenter clairement auparavant.

---

[17] Nattiez, Jean-Jacques, "trois modèles linguistiques pour l'analyse musicale", *Musique en jeu*, n°10, 1973, pp.3-11.

[18] Bayle, François, *musique acousmatique, propositions... ... positions*, Paris, Buchet/Chastel/Ina-GRM, 1993, pp. 93-152.
et
Couprie, Pierre, *La terminologie du genre éléctroacoustique*, Mémoire de DEA sous la direction de J.Y.Bosseur, Université de Paris IV-Sorbonne, 1998, pp. 62-65

[19] Collectif, *La musique électroacoustique*, Paris, Ina-GRM/Hyptique, 2000, CD-Rom Mac et PC.

Références

(Collectif) *Les Unités Sémiotiques Temporelles, éléments nouveaux d'analyse musicale*, Marseille, MIM, 1996, 96 pp.

Bayle, François, *musique acousmatique, propositions... ... positions*, Paris, Buchet/Chastel/Ina-GRM, 1993, 270 pp.

Bent, Ian, Drabkin, W., *L'analyse musicale. Histoire et méthodes*, Paris, Main d'Œuvre, 1998, 306 pp.

Brech, Martha, *Analyse Elektroakustischer Musik mit Hife von Sonagrammen*, Frankfort, Peter Lang GmbH, 1994, 211 pp.

Chion, Michel, "Du son à la chose. Hypothèses sur l'objet sonore", *L'analyse musicale*, n°11, 1998, pp.52-58.

Chion, Michel, *Guide des objets sonores, Pierre Schaeffer et la recherche musicale*, Paris, Buchet-Chastel/Ina-GRM, Bibliothèque de recherche musicale, 1983, 187 pp.

Couprie, Pierre, "Three analysis models for *L'oiseau moqueur*, one of the *Trois rêves d'oiseau* by François Bayle", *Organised Sound*, vol. 4, n° 1, avril 1999, pp. 3-14.

Couprie, Pierre, *La terminologie du genre éléctroacoustique*, Mémoire de DEA sous la direction de J.Y.Bosseur, Université de Paris IV-Sorbonne, 1998, 114 pp.

Couprie, Pierre, Battier, Marc, "L'*Acousmographe* : un outil pour l'analyse informatique de documents sonores", *Les cahiers de l'O.M.F.*, n°4, Paris, Université de Paris IV-Sorbonne, 2001, en cours de publication.

Delalande, François, "L'analyse des musiques électroacoustique", *Les musiques électroacoustiques, Musique en jeu*, n°8, 1972, pp.50-56.

Delalande, François, "La musique électroacoustique, coupure et continuité", *Ars Sonora*, n°4, 1996, publication en ligne.

Delalande, François, "Music Analysis and Reception Behaviours: *Sommeil* by Pierre Henry", *Analysis of Electroacoustic Music, Journal of New Music Research*, vol. 27, n°1-2, Dirk Moelants editor, juin 1998, pp.13-66.

Nattiez, Jean-Jacques, "trois modèles linguistiques pour l'analyse musicale", *Musique en jeu*, n°10, 1973, pp.3-11.

Payri, Blas, *Perception de la voix parlée : cohérence du timbre du locuteur*, Thèse de Doctorat, Université de Paris-Sud, 2000, 313 pp.

Roy, Stéphane, "Analyse des œuvres acoustiques : quelques fondements et proposition d'une méthode", *Electroacoustique-Québec : l'essor, Circuit, Revue nord américaine de musique du*

XX° siècle, vol. 4, n°1-2, Montréal, Les Presse de l'Université de Montréal, 1993, pp. 67-91.

Schaeffer, Pierre, *Traité des objets musicaux*, Paris, Le Seuil, Pierres Vives, 3/1977, 712 pp.

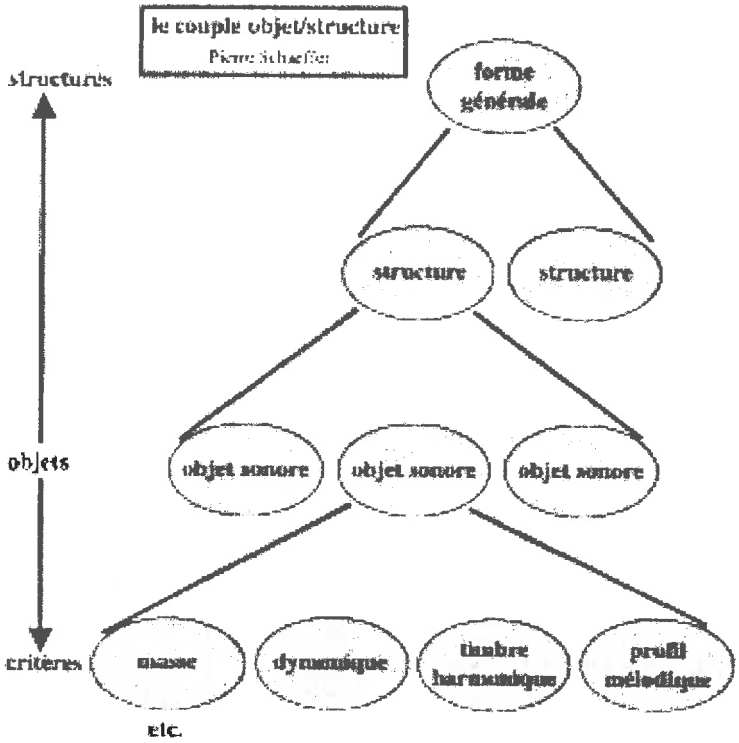
Schaeffer, Pierre, Reibel, Guy, Ferreyra, Béatriz, *Solfège de l'objet sonore*, Paris, Ina-GRM, 2/1998, trois compacts disques et un livret de 173 pages.

Schafer, Murray, *Le paysage sonore*, Paris, J.C.Lattès, 1979, 391 pp.

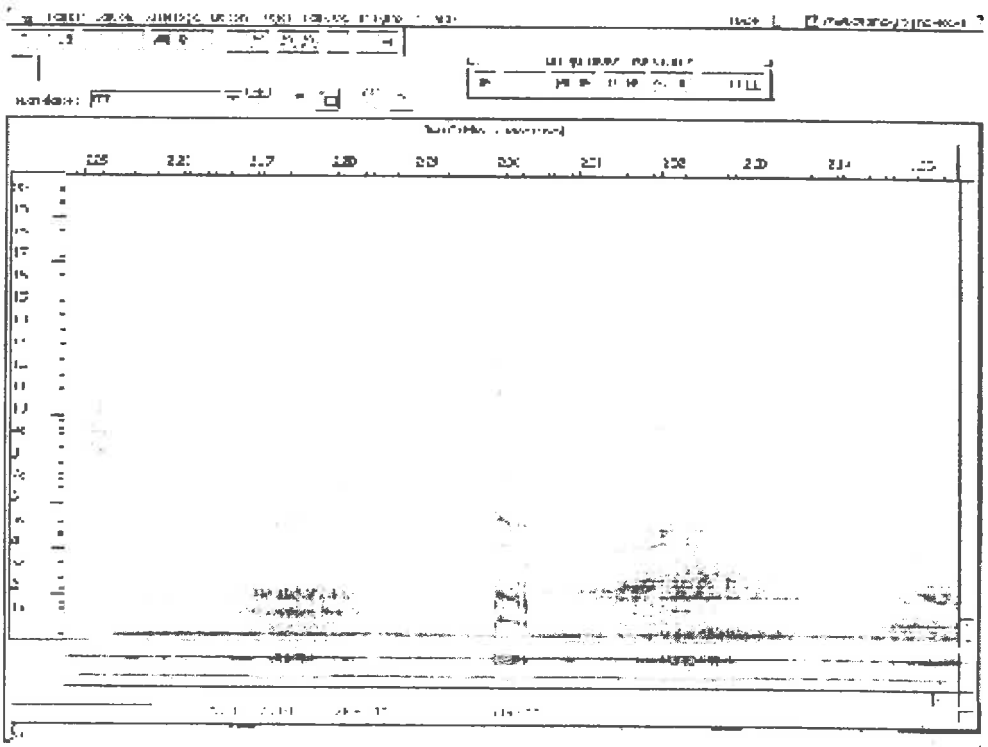
Smalley, Denis, "Spectro-morphology and Structuring Processes", *The Language of Electroacoustic Music*, Londre, Simon Emmerson/The Macmillan Press LTP, 1986, pp. 61-93.

Thoressen, Lasse, "Auditive Analysis of Musical Structures. A summary of analytical terms, graphical signs and définitions", *ICEM Conference on Electro-acoustic Music*, Stockholm, Royal Swedish Academy of Music, 1985, pp.65-90.

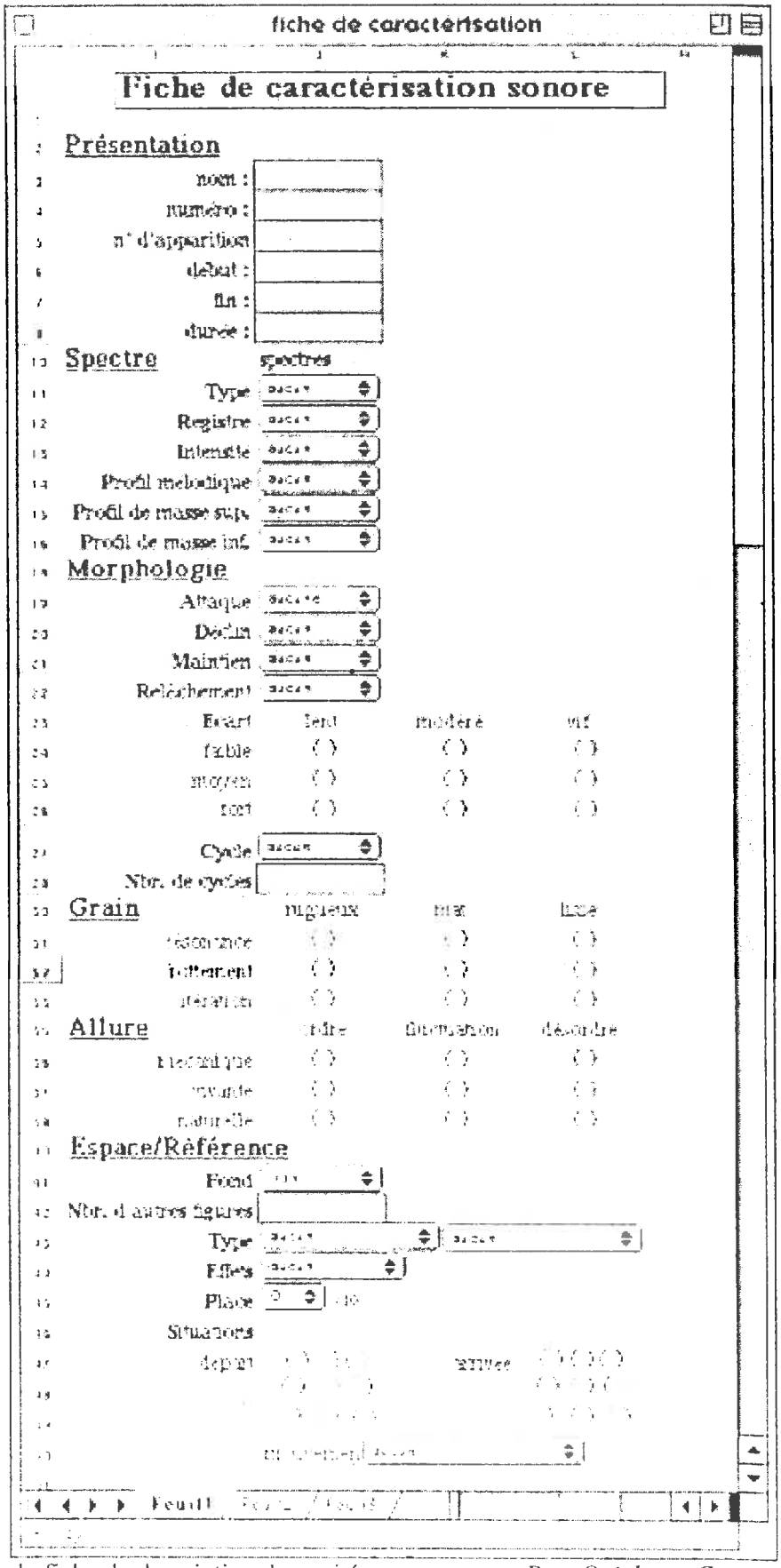
Annexe



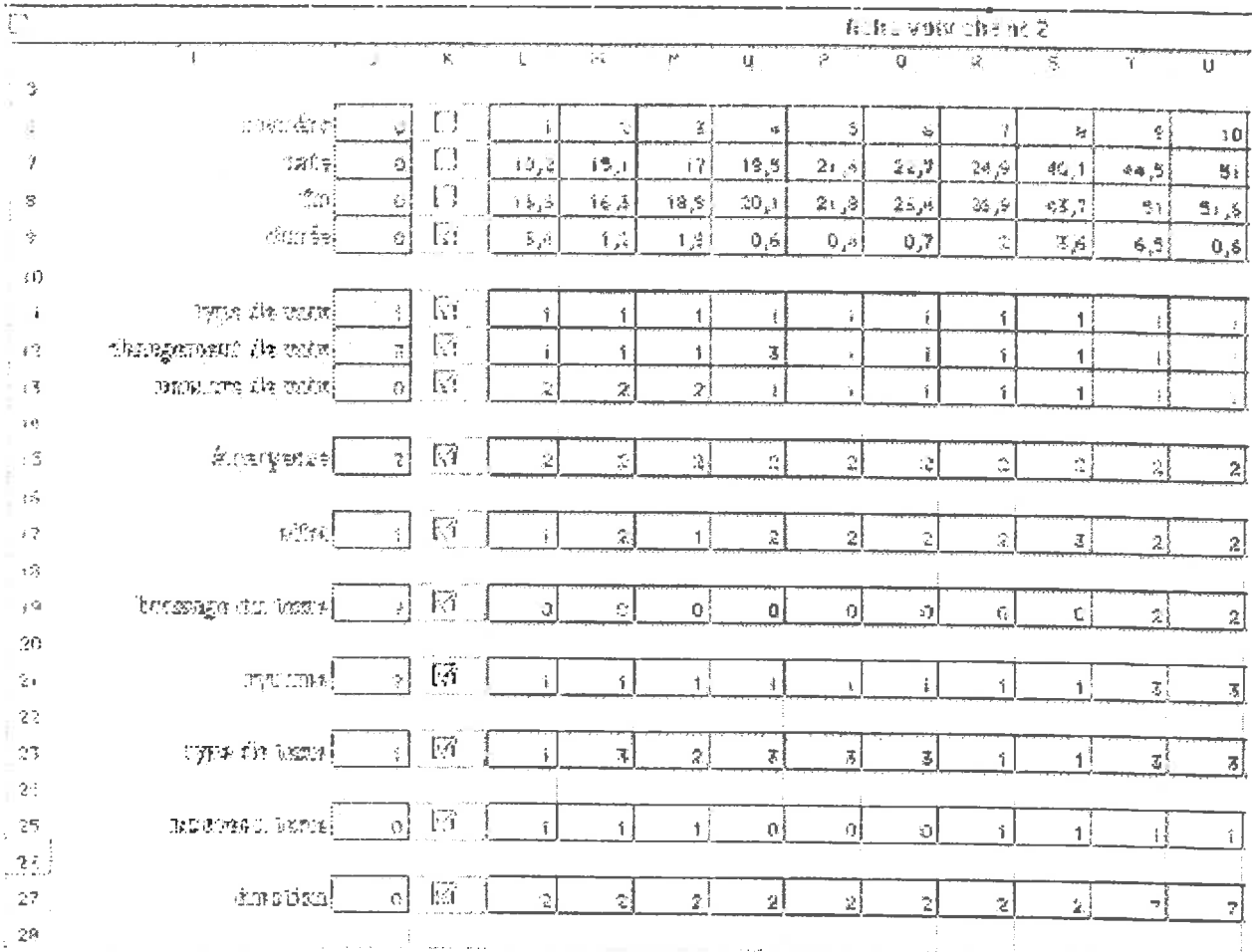
graphique 1 : le couple objet/structure selon Pierre Schaeffer



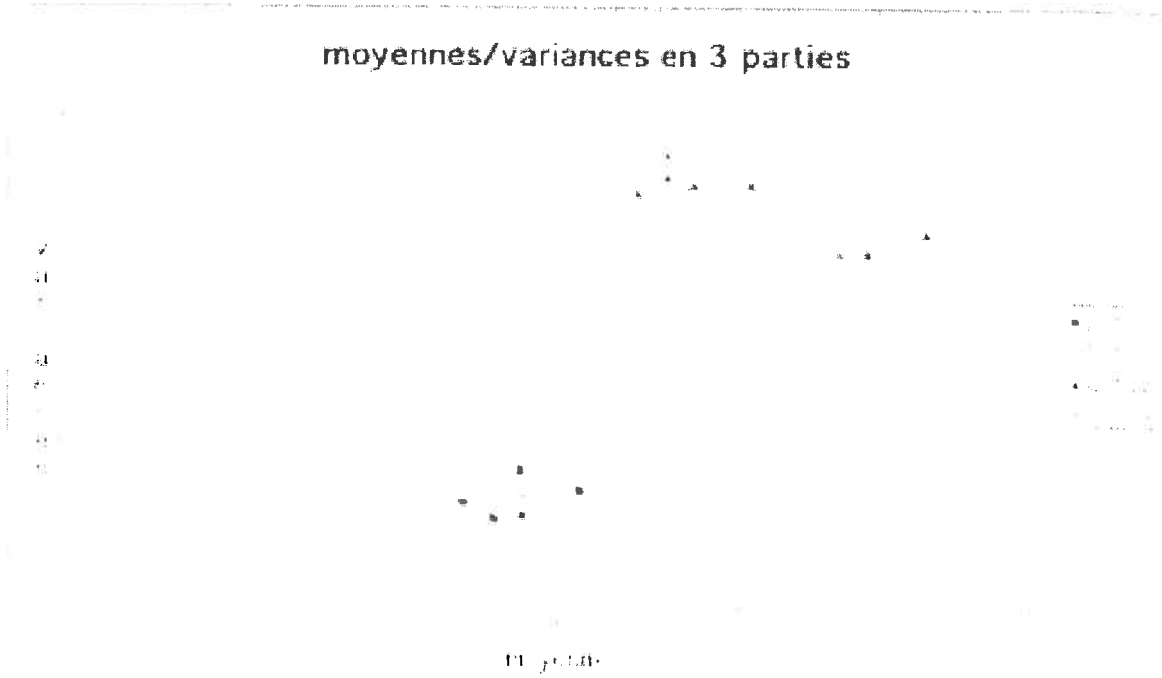
graphique 2 : un sonagramme réalisé sur l'Acousmographe



graphique 3 : la fiche de description des unités sonores pour *Don Quichotte Corporation* d'Alain Savouret



graphique 4 : la fiche de description des unités sonores pour le *Cantique des Cantiques* de Jacques Lejeune



graphique n°5 : représentation des moyennes et variances sur certains critères vocaux du *Chant n°3* du *Cantique des Cantiques* de Jacques Lejeune

THE NTA PROJECT

Giovanni Grosskopf
Agon Studio, Milano, Italy
Didier Guigue
Departamento de Música, Universidade Federal da Paraíba, João Pessoa, Brazil
ggrossk@tin.it , dguigue@openline.com.br

Summary: The GMT [1] is developing analytic tools for 20th century acoustic music. The theoretical background of these tools is the Object Oriented Analysis (OOA) methodology, which is based on the concept of sonic object. After shortly explaining this concept and methodology, we outline the GMT software project as a whole, then introduce the NTA software, and describe its tools with the help of musical examples.

Keywords: *Computer-assisted statistical analysis; sonic object analysis; 20th century music analysis; Max.*

SONIC OBJECT ANALYSIS

In the domain of acoustic music, a *sonic object* may be defined as the combination and interaction of the musical primary components (pitches) with the so-called secundary or statistical components, such as intensities, densities, and, generally speaking, space (achronic) and time (diachronic) filling up. It is a medium-level structure, between the lower level (pitch classes), and the upper level (macro-structure) [see fig. 1]. The way these objects are linked at this medium level happens to be a very important vector of form in 20th century music, as *timbre* becomes as structurally functional as pitch-classes were in older music or in some serial music.

In the background of the GMT [1] software project is the *Object Oriented Analysis* (OOA) methodology, which is thoroughly described in (Guigue 1996a, 1997a, 1997b). Very shortly and roughly speaking, the method consists in

1) segmenting the whole musical piece in a sequence of sonic objects [2];
2) describing the structure of each object according to a selection of relevant statistical components, and
3) quantifying the gap of sonic continuity between consecutive objects as a whole, or for

---

[1] GMT (*Grupo de Pesquisas em Música, Musicologia e Tecnologia Aplicada*) is a research Group coordinated by Didier Guigue at the Music Department of Universidade Federal da Paraíba (Brazil), with a grant from CNPQ (Brazilian Council for Research). Its members are researchers, composers, graduate and post-graduate students in music and computers, from various Brazilian universities. Giovanni Grosskopf (Agon Studio, Milano, Italy) is an associated researcher.

[2] The method admits polyphony of objects — i.e. multi-layered sequences.

each component, or for homogeneous groups of components [3]. This quantification configures a crucial aspect of the piece's formal *kynesis* and allows the form to be inferred from the succession of more or less contrasted sonic objects.

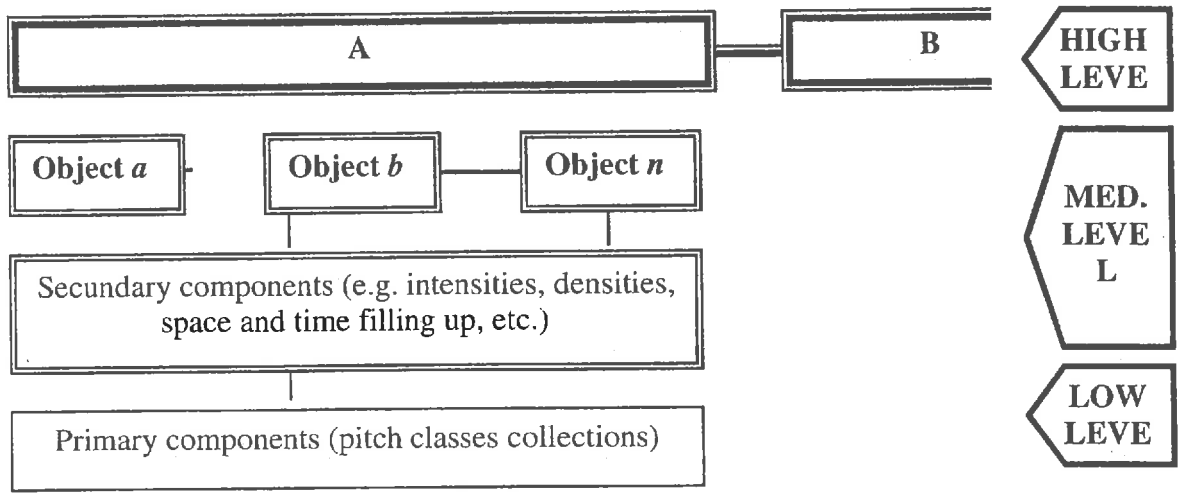


Fig. 1. The three levels of a musical structure, according to OOA model. The sonic object, at the medium level, is a combination and interaction of primary and secundary components. The high level is formed by the sequence of macro-sections A, B, ... N.

The basic rule for evaluating how important is a given component in the design of the sonic structure of a given object, is the rating of its relative *complexity*. The maximum "complexity" corresponds to the configuration that would produce a sonority as "complex" as possible. The meaning of the term "complexity" varies according to the nature of the component. When dealing, for instance, with the number of simultaneous notes (density) in a given object, the maximum value is that corresponding to a fully saturated object (e.g. a *cluster*). On the other side, a hypothetical empty object would receive a density weight of zero. It must also be observed that the concept of saturation (in the sample case of the density) will vary from one object to the other, according to its own boundaries (its lowest and highest pitches). Therefore, as a general rule, *all* quantifications must be relative (e.g. from 0 to 1), for it is the only way to compare and process together heterogeneous components, to compare objects, and, finally, to infer technical, aesthetic or stylistic conclusions from a piece or a set of pieces.

The GMT Software Project

All these step and rules will be implemented in a future stand-alone application we are working on at GMT. The main concern for this software to be fully functional is to implement IA algorithms for segmenting a whole musical piece into a number of units — the sonic objects —, and infer (or suggest) the relevant components to be applied (Trajano *et al.* 2000). By the meantime this segmentation is manually done, applying the

[3] If relevant for the analytic purpose, components can be grouped in a hierarchical structure.

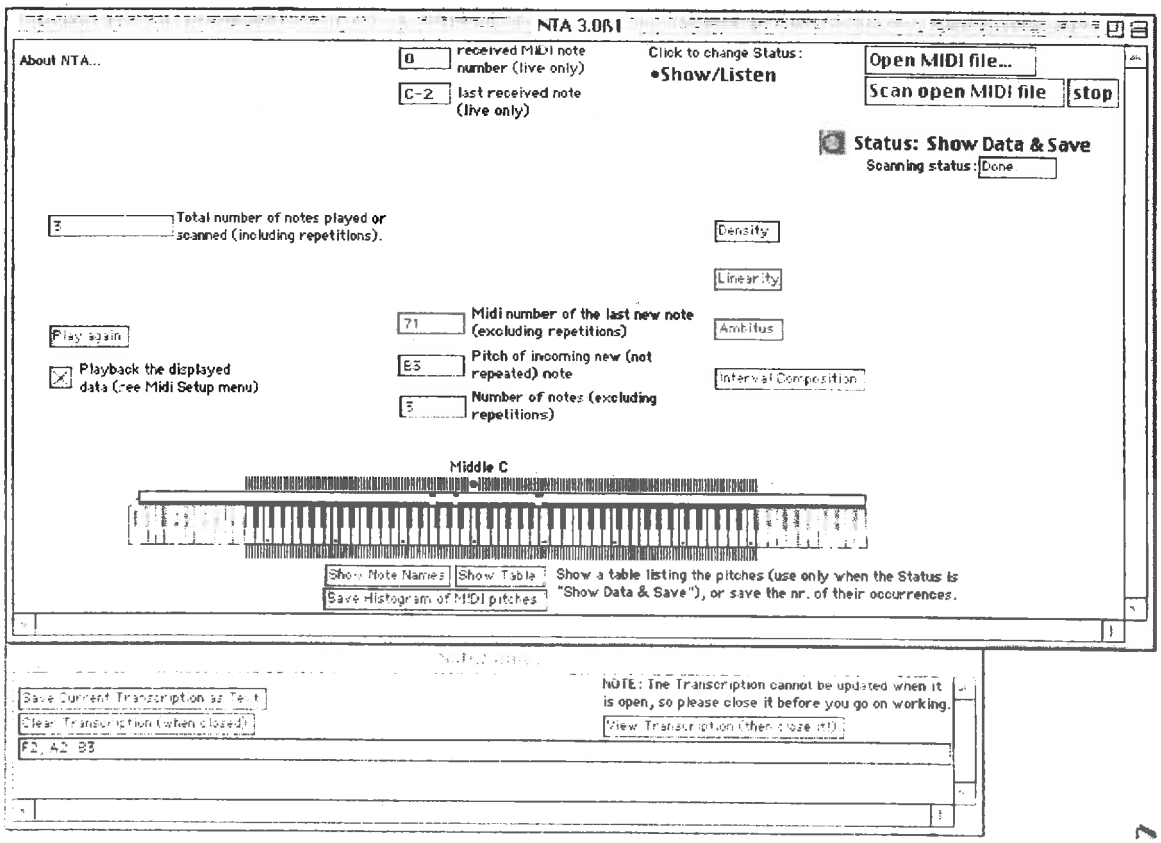
OOA rules to the written score of the music [4].
On another hand, we are developing simultaneously two sets of analytic tools:

- the NTA (*NonTonalAnalysis*) software, developed by Didier Guigue and Giovanni Grosskopf on Max; and
- SOAL, a library of tools developed by Didier Guigue and Ernesto Trajano, the first version of which is about to be released for the Patchwork/OpenMusic community [5].

As long as the automated segmentation is not concluded, each segment of the score representing a sonic object must be encoded as a single Midi file, in both software packages.

NonTonalAnalysis

The present stage (version 3.0) [6] of the *Non-Tonal Analysis* (NTA) software project involves the development on Max (for the Macintosh) of a software capable to integrate as many as possible of the functions of our previous works [Grosskopf 1998, 1999] in a unique environment, providing also a friendly user interface.



[4] The focus on the written aspect of a music which is intended to be analyzed in its sonic properties is surely a point to discuss, but this is not the scope of this paper.

[5] A preliminary set of Patchwork abstractions is available at the software section of the GMT site: http://www.liaa.ch.ufpb.br/~gmt . Some of them was described in [Guigue 1996b].

[6] The version 2.0 of NTA is available at http://space.tin.it/musica/ggrossko .

Fig.2. The main window of NTA with the basic elements of its user interface.

NTA acquires the data by opening and scanning a standard midi file (representing the sonic object to be analyzed), or by playing the desired notes on a connected midi instrument. The interface looks as in fig.2.

The algorithms already implemented can perform the following tasks:

- count notes and show which notes are present in a played music object or in a midi file, which have been repeated or not repeated
- show how many occurences of each note are there, in the form of a Histogram
- evaluate the Density of the object
- evaluate the Linearity of the object
- evaluate the Ambitus of the object
- evaluate its Interval composition

Apart from the first task, which has a quite generic use, all the other parameters are analyzed and shown within their own subwindows, which are opened by clicking on the corresponding button on the right part of the main window. The last four parameters deal, exclusively, with the achronic structure of the sonic object. That means NTA considers any input object as a chord [7].

All what follows will be illustrated taking into account that the development of NTA is carried on using the programming environment Max (by Miller Puckette and David Zicarelli) for the Macintosh [Zicarelli *et al.* 1990]. Max, an application originally conceived for real-time midi and sound elaboration during live electronics concerts, can therefore be successfully employed also for data analysis and musicology, owing to its powerful capabilities of interacting with MIDI objects and with the user in a friendly and transparent way, and owing to the many ready-made objects for this purpose which are included in its package (like the *Histo* object, which provides the Histogram).

Density

Density is a measure of the ratio between the number of notes of the object and the maximum number of notes that could be comprised between its extreme pitches in a given musical system (here, the chromatic tempered scale). Thus, an object like a chromatic cluster will have a maximum density value. To implement this algorithm requires first to detect the number of notes of the object, then to detect the extreme (lowest and highest) pitches in the object, in order to complete the calculation, a task to be implemented also in many other algorithms.

Linearity

Linearity (fig.3) measures how far is the object from being composed all of equal intervals. In this case it would be a linear object (like a whole-tones hexatonic scale or an

[7] Time-related tools are implemented in the SOAP library.

augmented triad). The possible degrees and variations of linearity depend not only on the extension of the object (the distance between its lowest and its highest note), but also on the number of notes that are contained in its extension. This is a matter of degrees of freedom, and leads to use two different measures of linearity: an absolute one and a relative one.

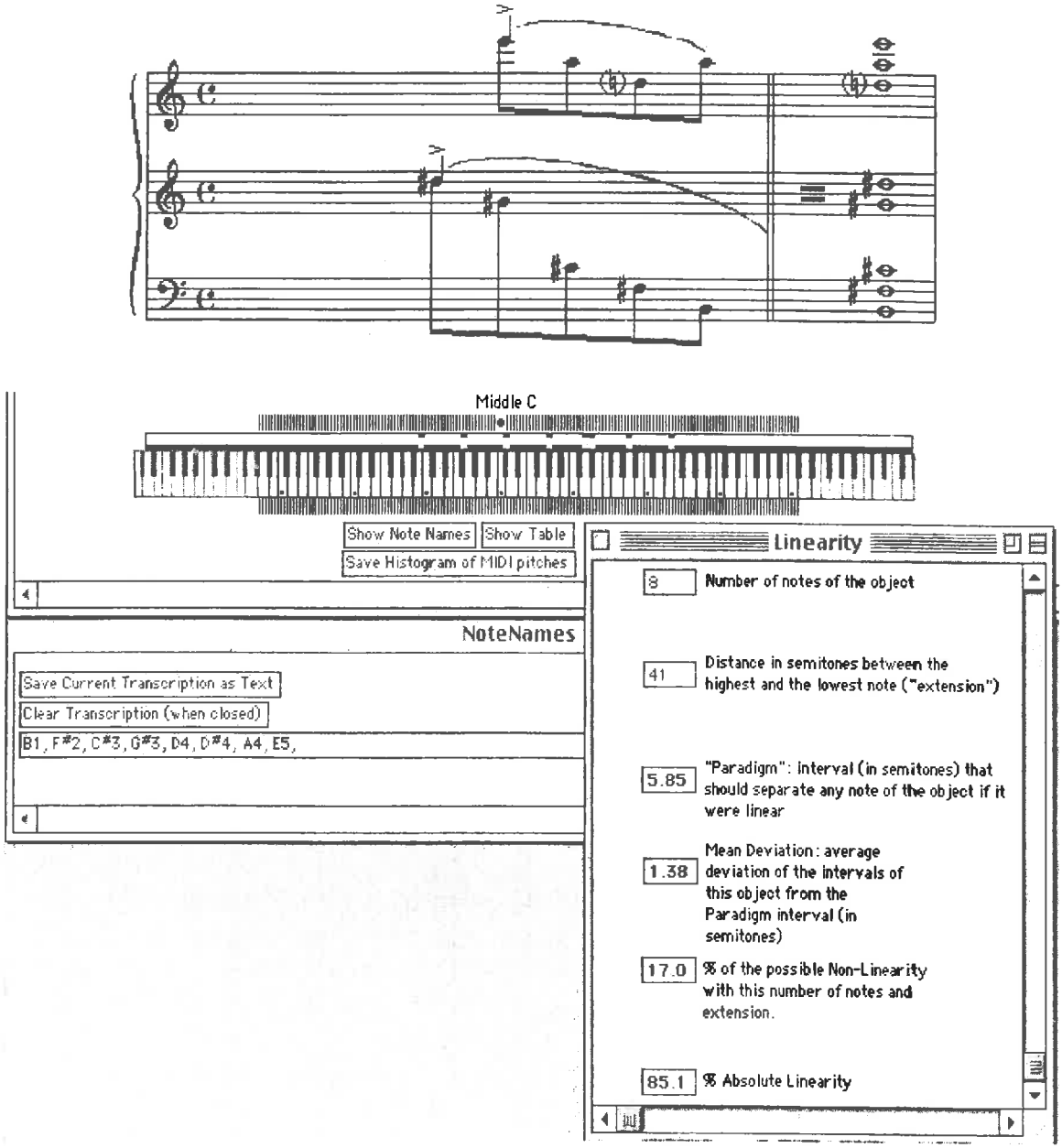


Fig.3. An example of analysis in the Linearity subwindow.
(György Ligeti, Études pour piano, premier livre, Étude 2, bar 5)

To perform Linearity, first NTA calculates the "Paradigm", the ideal interval (in semitones) that should separate any note of the object if it were linear. It is obtained

simply by dividing the extension of the object (the distance between its highest and lowest notes) by the number of consecutive intervals in the object.
The "relative" Linearity measure is then defined as the ratio between the Standard Deviation of the intervals of this object from the Paradigm interval (in semitones), and the Standard Deviation, from the Paradigm, of the intervals of an object with the same number of notes, corresponding to the case of the highest possible non-linearity (all the intervals but one = 1 semitone, and the remaining one = the largest possible interval).
The "absolute" Linearity measure is the same, but takes into account the number of vacant places (the chromatic steps not occupied by notes) actually present in the object, compared to the number of vacant places that would be the maximum possible if this object had only 3-notes [8].

Ambitus

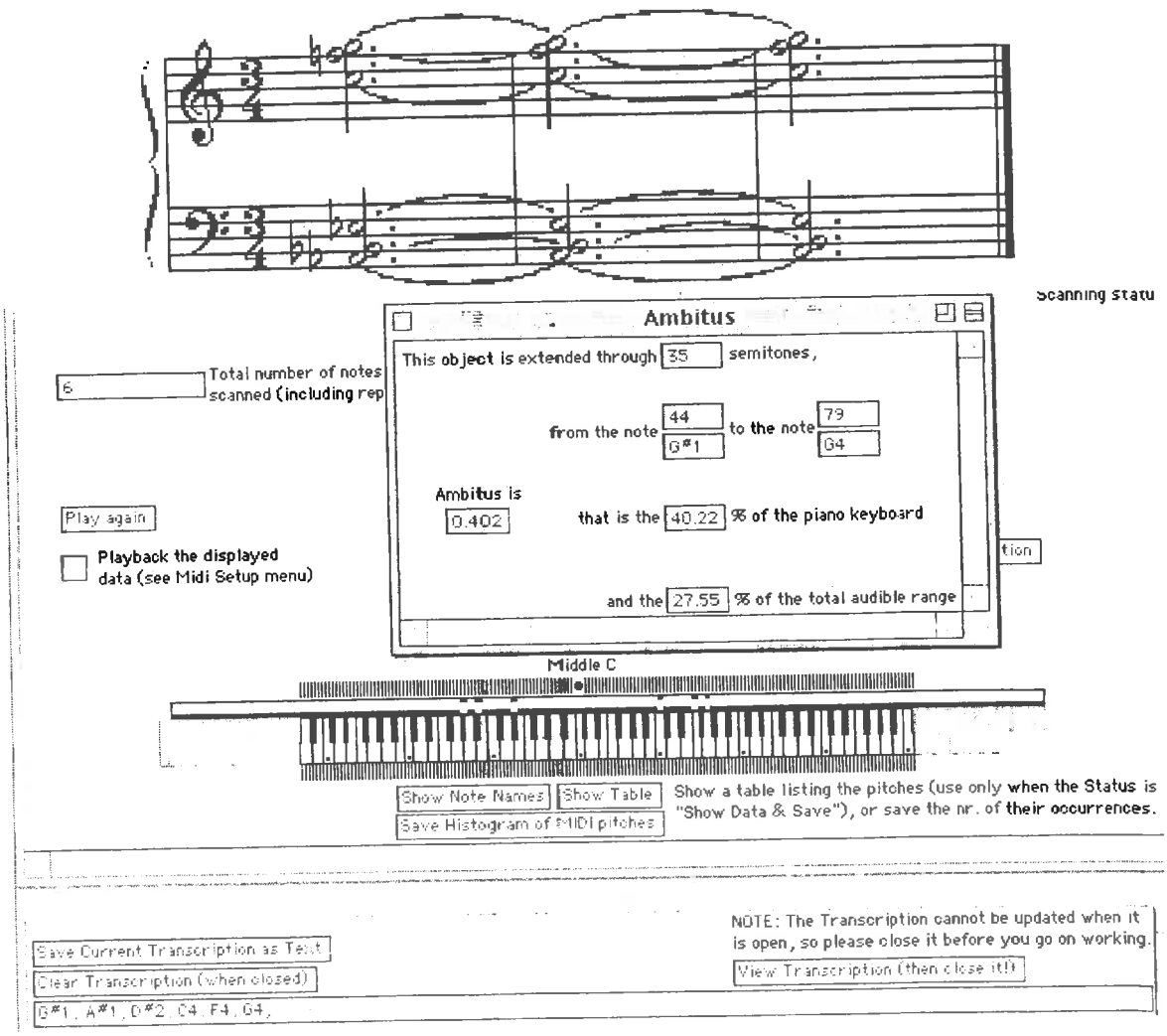


Fig.4 An analysis example in the Ambitus subwindow.

[8] In the case of triads, the two measures, absolute and relative, are one the complement of the other (so that their sum gives 100).

(Niccolò Castiglioni, Come io passo l'estate, piano suite, end of the first movement)

Ambitus (fig.4) evaluates the extension of the object (the distance between its lowest and its highest pitch), and compares the result with some common standards of comparison, like the extension of a piano keyboard (88 notes, that is 87 semitones) or the average audible range (the latter has been approximated to the total range of the MIDI notes, 127 semitones). Other comparison standards could be added.

Interval Composition

The goals of the Interval Composition algorithm (fig.5) are:
- to list all the existing intervals formed by any couple of pitches in the object
- to find how many occurrences of each existing interval are in the object
- to list which pitches form them
- to list their position within the object (e.g. if they are the lowest note, or the second one, or another inner note, or the highest one).

NTA is capable to measure the intervals in three ways:
- Without any octave reduction, having the semitone as the measure unit.
- With a reduction within the compass of one octave, using the usual standard names for the intervals.
- With a reduction within the compass of one major ninth, using the usual standard names for the intervals. This method has been implemented to support the Interval Perception algorithms (see below).

Besides detecting all the intervals, NTA will detect also some combinations of three notes, the sonority of which is not merely the result of the sum of the sonorities of the intervals composing them, but has a recognizable timbre in itself, and can have a large influence on the sonority of the whole object which contains them. To detect these three-notes combinations will be particularly useful for the Interval Perception algorithms (see later). The combinations to be detected are the following:
- major triads, in any position or inversion
- minor triads, in any position or inversion
- augmented triads (like C, E, G sharp), also in the open disposition (E, C, G sharp)
- the so called "conflicts of thirds", that are all the situations occurring when a major third (or minor sixth) and a minor third (or major sixth) are both present and share a common note, but do not form a triad, rather suggesting, from the point of view of traditional tonality, a minor and a major triad with the same root tone. For instance, C, E flat, E natural (that could suggest a conflict between C major and C minor) or C, E flat, C flat (that could suggest a conflict between Ab major and Ab minor: here the root tone itself is not included in the note combination).
- in some cases, the combinations traditionally analyzed as uncomplete chords of dominant seventh (like F, G, B).

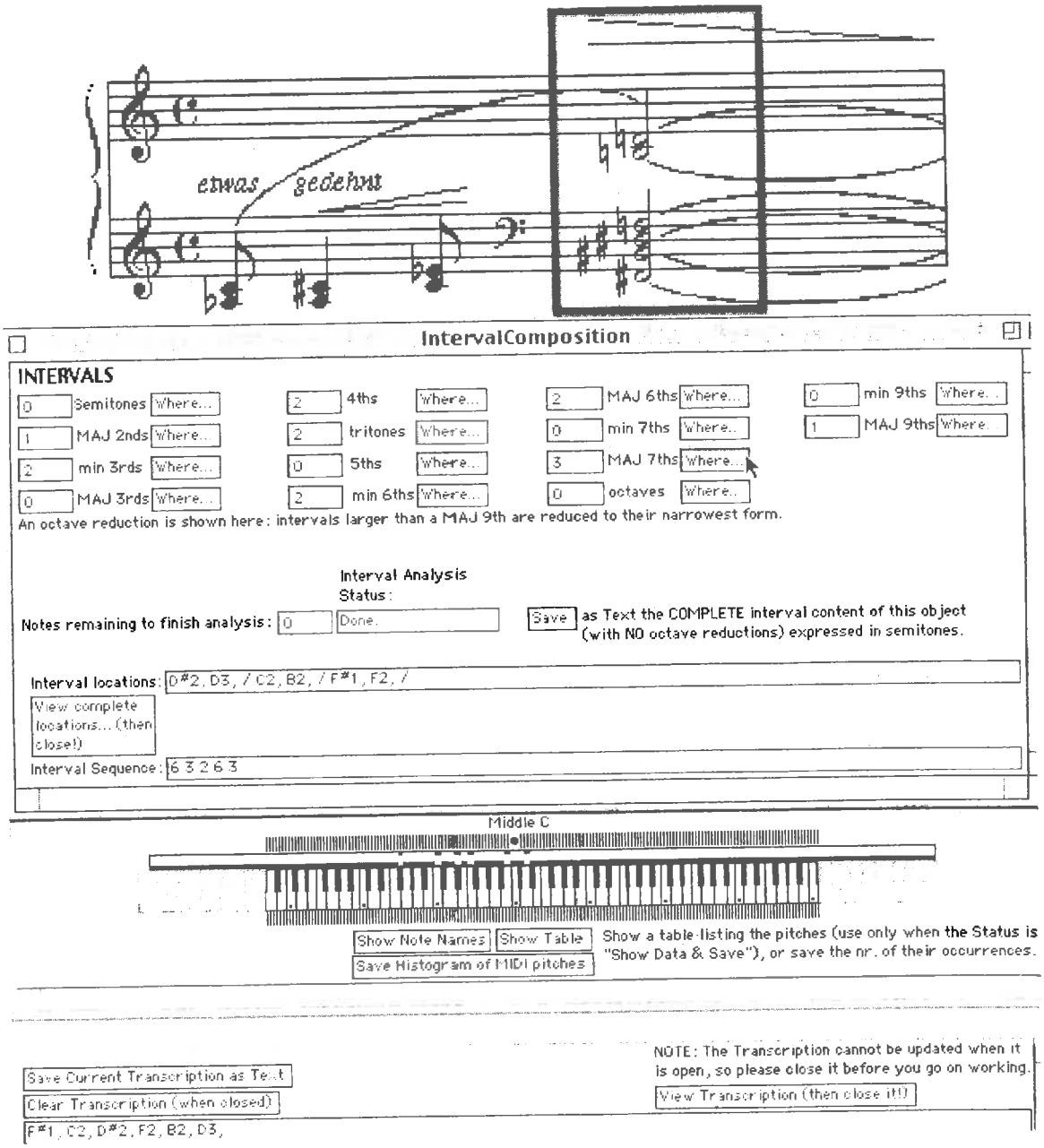


Fig.5. An analysis example in the Interval Composition subwindow.
(Arnold Schoenberg, op.19 n.2, bar 6)

Some future developments

In the near future some Interval Perception algorithms will be added to NTA, which will allow to detect which ones of the intervals composing an object are perceived more clearly and have the greatest influence on its overall sonority, depending on their nature (strong consonances and strong dissonances are perceived better by the human ear than the other intervals) and on their position in the object (for instance, the lower notes of a chord have usually a greater influence on its sonority than the upper ones). This will allow to classify the objects (mainly non-traditional, non-triadic, non-tertian chords), according to their sonority and to the "composition" of their sonority, providing a sort of "interval spectrum" of a harmonic object, which will list all the intervals contained in the object in hierarchical order, according to their importance in contributing to the global sonority of the object itself. This will allow to detect the similarities or dissimilarities in the sonority of different objects, and would be particularly useful in the field of computer-assisted composition.

Another important development will be the implementation of several algorithms (according to different approaches) to measure the dissonance level of an object. We have listed many methods for the measure of dissonance , and it's our goal to implement the largest possible number of them in NTA, in order to provide a comparison among them when they are applied to the analysis of the same object.

A MAX feature requiring attention

A major and not obvious problem with all the Max algorithms is that all the calculation must be triggered by a timed impulse (called the *bang*), that is, the order to perform the calculation must be given at a certain precise moment, and, owing to the particular configuration of Max, this moment must often be controlled by time-related objects (delays, timers, which count milliseconds, and counters, which count the occurrences of an event), and not by data-checking objects (objects which could check if certain data have already been produced or not). However, the exact instant in which Max *outputs* the data from the algorithm which *precedes* the one we are working on is sometimes quite unpredictable (for instance: if we open a very large midifile and transmit its contents from one module of Max to another, sometimes we cannot know exactly how many milliseconds will be necessary for all the data to reach the new module). The problem is therefore that, at the moment in which we would like to start the calculation of the new algorithm, we are not always sure that all the necessary data are available for the calculation to be performed, because they may not have been all produced or detected. This requires a particular care, and the opposite risks are either to slow down an algorithm in order to be sure that all it has all the necessary time to accomplish all its tasks and retrieve all the necessary data, or to have incorrect results with large data structures (like large midifiles), because a calculation starts when not all the necessary data are available yet. Besides, some events produce by themselves a *bang* and start the calculations anyway. The risk of triggering a calculation too early or too late must therefore be considered attentively.

REFERENCES

[Grosskopf 1998]: Grosskopf, Giovanni. "Un approccio diverso all'uso dell'armonia atonale ed all'analisi di accordi non tonali tramite il software "NonTonalAnalysis"", Proceedings of the Italian National Conference of Electronic Music "La Terra Fertile" 1998, Istituto Gramma, L'Aquila, 1998.

[Grosskopf 1999]: Grosskopf, Giovanni. "NonTonalAnalysis: a different approach to the analysis of atonal chords: achieving a clearly perceivable directional logic in atonal harmony.", in Proceedings of the International Conference "The Principles of Musical Compositions" held in 1999, Lithuanian Academy of Music - Lithuanian Composers Union, Vilnius, 2001.
http://www.liaa.ch.ufpb.br/~gmt/Grossk/GKpapers/LITPIC.htm

[Guigue 1996a]: Guigue, Didier. "Principes méthodologiques d'une analyse orientée objets de la musique du XXe siècle". Paris, Ircam, Médiathèque Online. http://mediatheque.ircam.fr/articles/textes/Guigue96a/.

[Guigue 1996b]: "Un environnement informatique pour une approche analytique orientée objets de la musique du XXe siècle", in ASSAYAG, M., CHEMILLIER, M., & ELOY, Ch. (eds), 1996 : Troisièmes Journées d'Informatique Musicale JIM 96, Cahiers du GREYC, Vol. 4, pp. 266-272
www.ircam.fr/equipes/repmus/jim96/actes/guigue/guigue.html

[Guigue 1997a]: Guigue, Didier. Une Etude "pour les Sonorités Opposées" - Pour une analyse orientée objets de l'oeuvre pour piano de Debussy et de la musique du XXè siècle. Villeneuve d'Asq, Presses Universitaires du Septentrion, 559 p.

[Guigue 1997b]: Guigue, Didier. "Sonic object : a Model for Twentieth Century Music Analysis". The Journal of New Music Research, Vol. 26 n. 4, pp. 346-375.
http://www.swets.nl/jnmr/vol26_4.html.

[Trajano *et al.* 2000]: Trajano, Ernesto. Guigue, Didier. Ferneda, Edilson. ."Segmentação automática de fluxos musicais: uma abordagem via agentes racionais", Revista Eletrônica de Musicologia, Vol. V.2, 12/2000.
http://www.cce.ufpr.br/~rem/REMv5.2/vol5.2/guiguetrajano/guiguetrajano.htm.

[Zicarelli *et al.* 1990]: Dobrian, Christopher and Zicarelli, David. "MAX, An Interactive Graphic Programming Language", Menlo Park, California, Opcode Systems – IRCAM, 1990.

Un *pitchtracker* monophonique

Damien Cirotteau Dominique Fober Stephane Letz Yann Orlarey

Grame - Département Recherche
9,rue du Garet BP 1185
69002 LYON CEDEX 01
Tel +33 (0)4 720 737 00 Fax +33 (0)4 720 737 01
(cirotteau,fober,letz,orlarey)@grame.fr

Résumé

Nous présentons ici un détecteur de hauteur de note basé sur une amélioration du vocodeur de phase. Cette amélioration, permettant une meilleure précision en temps et en fréquence, est parfaitement adaptée à une utilisation en temps réel. Une attention particulière a été portée sur la possibilité d'intégration de ce détecteur dans différents systèmes.

1 Introduction

La musique interactive nécessite des outils de déclenchement en temps réel. La majorité des détecteurs de hauteur de notes est basée sur une analyse des harmoniques du signal impliquant généralement une transformée de Fourier. Il en résulte le traditionnel compromis temps contre fréquence. En effet, la précision en fréquence de l'analyse est proportionnelle au nombre de points du signal discrétisé que l'on utilise pour cette analyse. Le vocodeur de phase est un moyen classique de minimiser ce compromis.

Une amélioration originale du vocodeur de phase utilisant la dérivée du signal a été réalisée dans [1] afin d'affiner la précision de la mesure. Ce vocodeur de phase nous permet d'obtenir la fréquence des différentes composantes d'un signal avec une excellente précision compte tenu de la taille du buffer d'analyse.

Une fois que nous avons les composantes du signal, il faut extraire la fondamentale. Cela revient à trouver la périodicité des composantes du signal. Une fonction de *maximum likelihood* permet de déterminer le meilleur candidat parmi les composantes du signal et éventuellement extrapole la fondamentale même si elle ne fait pas partie des fréquences détectées précédemment.

Enfin, il faut transformer la fréquence en une information directement compréhensible par un système extérieur. La norme Midi a donc tout naturellement été choisie.

Lorsque une note stable est détectée, le *pitchtracker* fourni une information de début de note, puis suit l'évolution de cette note au cours du temps : modulation de hauteur ou d'amplitude. Lorsque la note passe sous un certain seuil d'amplitude ou dépasse un seuil de hauteur paramétrable, une information de fin de note est renvoyée.

Le détecteur que nous avons mis en place n'est pas un interface Midi proprement dite : il n'envoie aucune commande Midi lui-même mais fournit les informations au système extérieur qui se charge des messages Midi. Le *pitchtracker* est fournit sous forme de librairie C. Notre

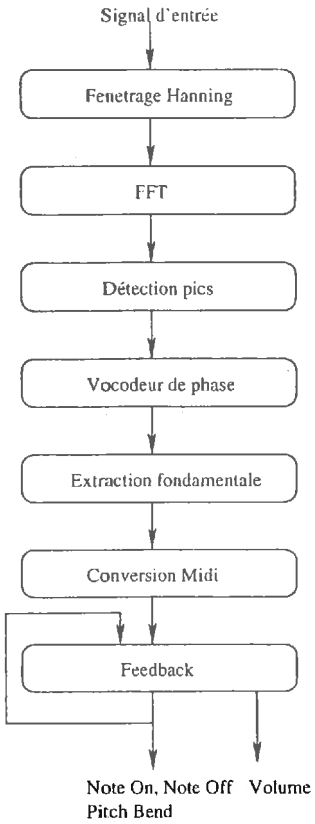


FIG. 1 – Principe de la détection

détecteur ne fonctionne pas de manière autonome et doit être intégré dans un système global. Par analogie notre système correspondrait à la partie mécanique d'un clavier numérique alors que le système hôte prendrait en charge la partie électronique.

2 Obtention des harmoniques

2.1 Récupération des pics significatifs

Il faut tout d'abord extraire du signal les informations fréquentielles pertinentes. Nous réalisons une FFT N points du signal pour obtenir son spectre. Le signal aura préalablement été fenêtré par une fenêtre de Hanning.

La recherche des maxima locaux nous donne les composantes sinusoïdales du signal. En plus d'être un maximum local, un pic doit être supérieur à un certain seuil pour être significatif. Typiquement le seuil est compris entre 1 et 5% du maximum du spectre.

Cependant, la précision fréquentielle de chaque composante ainsi obtenue est proportionnelle à la fréquence d'échantillonnage R et au nombre de points de la FFT. Ainsi,

$$f_p = m_p \frac{R}{N} \tag{1}$$

avec m_p le numéro du point correspondant au pic dans le spectre et f_p la fréquence de la composante sinusoïdale. Pour une fréquence d'échantillonnage de 44,1 kHz et une FFT de 512 points par exemple, l'écart entre deux points successif dans le spectre est de 86,13 Hz.

Nous avons donc une incertitude très grande sur la fréquence d'un pic. Nous allons voir qu'il est possible d'obtenir la fréquence exacte de ce pic à partir de cette FFT et de la FFT du signal dérivé.

Cependant, il est très important de noter que l'écart entre deux pics successifs donc entre deux harmoniques d'une note doit être supérieur à $\frac{N}{R}$ et donc il est nécessaire d'adapter la taille de la FFT à la hauteur de la note à détecter. Les valeurs précédentes, par exemple, permettent en théorie de détecter des notes supérieures au Fa 1.

2.2 Vocodeur de phase amélioré utilisant la dérivée du signal

Myriam Dessainte-Catherine et Sylvain Marchand ont montré dans [1] et [2] qu'il était possible d'obtenir une bien meilleure résolution en fréquence qu'avec un classique vocodeur de phase. Le principe de base est que la dérivée d'un sinus est un sinus de même fréquence mais de phase différente. Comme un signal audio est la somme d'oscillateurs sinusoïdaux, la dérivée d'un signal audio est un signal de même fréquence mais de phase différente.

L'expression d'un signal audio est donnée par la formule suivante :

$$a(t) = \sum_{p=1}^{P} a_p(t)\cos(\varphi_p(t)) \tag{2}$$

avec

$$\frac{d\varphi_p}{dt} = 2\pi f_p(t) \tag{3}$$

i.e.

$$\varphi_p(t) = \varphi_p(0) + 2\pi \int_0^t f_p(u)du \tag{4}$$

La phase initiale étant quelconque et n'intervenant pas dans la mesure, nous pouvons arbitrairement la choisir nulle. De plus, nous considérons que l'amplitude et la fréquence de chaque composante varie très lentement par rapport à la taille de la fenêtre d'analyse. Leurs dérivées sont donc proches de zéro.

Il vient alors de (2) et (4) :

$$\frac{da(t)}{dt} = \sum_{p=1}^{P} 2\pi f_p(t) a_p(t)\cos(\varphi_p(t) - \frac{\pi}{2}) \tag{5}$$

Le signal que nous traitons étant discret, notons DFT^k le spectre de la DFT de la k-ième dérivée du signal.

$$DFT^k[m] = \frac{1}{N}\left|\sum_{n=0}^{N-1} w[n]\frac{d^k a}{dt^k}[n]e^{-j\frac{2\pi}{N}nm}\right| \tag{6}$$

avec w une fenêtre d'analyse de taille N.

Comme nous avons obtenu les pics lors de la précédente étape, nous avons une fréquence approximée (1) de chaque composante. Pour chaque pic p nous avons un maximum dans DFT^0et DFT^1. Cela nous permet de déterminer avec précision la fréquence des harmoniques.

$$f_p = \frac{1}{2\pi}\frac{DFT^1[m_p]}{DFT^0[m_p]} \tag{7}$$

Il est également possible, par la même méthode, d'obtenir l'amplitude de chaque partiel de manière précise mais cela est hors du propos de cet article.

3 Extraction de la fondamentale

Maintenant que nous avons obtenu les fréquences des composantes de façon précise, il s'agit de déterminer la fondamentale. Nous avons en entrée de cet algorithme un jeu de fréquences correspondant aux différentes composantes sinusoïdales du signal.

Une fonction du type *maximum likelihood* suivant les idées de [4] est utilisée. Pour chaque composante, calculons la valeur de la fonction suivante :

$$\Gamma(f) = \sum_{p=1}^{P} O_p(f) Y_p(f) \tag{8}$$

avec f la fréquence de la composante du signal sur laquelle nous effectuons la mesure, p les autres composantes du signal de fréquence $h(p)$ et P le nombre de pics trouvés dans le spectre.

Le facteur Y_p est une fonction non nulle à proximité d'une certaine fréquence. C'est une fonction triangulaire centrée sur nf.

$$Y_p(f) = \frac{nf - h(p)}{h(p) - f_{min}} + 1 \quad f_{min} \leq nf \leq h(p) \tag{9}$$

$$Y_p(f) = \frac{f_{max} - nf}{f_{max} - h(p)} \quad h(p) \leq nf \leq f_{max}$$

$$Y_p(f) = 0 \ sinon$$

avec $n \in [2, P]$.

Ce facteur représente la tolérance acceptée pour qu'une composante A soit considérée comme l'harmonique d'une composante B. Plus A sera proche d'un multiple B, plus la valeur de Y sera importante et si A est multiple de B alors Y est égal à 1. Si A n'est pas proche d'un multiple de B, alors Y sera nulle et A ne sera pas considérée comme "participant" à la note d'harmonique B. Comme nous réalisons un *pitchtracker* monophonique, nous avons choisi une tolérance d'un quart de ton.

Le facteur $O_p(f)$ est fonction de l'index de l'harmonique. Nous considérons que la contribution des harmoniques bas est plus importante que celle des harmoniques élevés. Il faut donc un coefficient reflétant la hauteur de l'harmonique. Soit :

$$O_p(f) = \frac{0.9}{i - 0.1} \tag{10}$$

avec i la valeur arrondie de $\frac{h(p)}{f}$.

La valeur de f qui maximise Γ est la fréquence de la fondamentale. Cependant, il existe des notes dont la fondamentale est absente du signal (et parfois même plusieurs harmoniques bas sont absents en plus de la fondamentale).

Il est possible, grâce à notre fonction de *maximum likelihood* de déterminer une fondamentale même s'il elle n'est pas présente dans le jeu initial des composantes sinusoïdales. Si f_0 est la fréquence maximisant Γ et que f_1, sous-multiple de f_0, renvoie une valeur de Γ plus grande alors f_1 devient la fondamentale du signal.

4 Conversion midi et feedback

Nous avons obtenu la fréquence de la fondamentale. Cette information n'est pas directement exploitable par un système interactif. Nous allons la convertir en information Midi.

A chaque note Midi correspond une fréquence donnée que nous appellerons valeur fréquentielle Midi. Cette fréquence est donnée par l'accord de l'instrument et les réglages du système. Typiquement, la note 60 correspond à 262 Hz soit Do3.

Pour chaque nouvelle mesure i, nous cherchons la note Midi la plus proche de la fondamentale trouvée au moyen d'une dichotomie. Nous déterminons aussi la valeur du pitch bend fonction de l'écart entre la fréquence vraie et la valeur fréquentielle Midi.

Comme le signal est continu et que nos buffers de mesures sont relativement petits, il est intéressant de prendre en compte les résultats de la mesure précédente et ceci pour différentes raisons. Soit i la mesure courante et $i-1$ la mesure précédente. Soit f la fréquence, M la valeur de la note Midi et P la valeur du pitch bend.

Si f_i est un multiple de f_{i-1}, il est possible que la détermination de la fondamentale lors de l'étape précédente est échouée. Nous vérifions alors, pour l'étape i, s'il existe ou non des pics de fréquence multiple à f_{i-1}. Si c'est le cas, f_i prend la valeur de f_{i-1} et nous mettons M_i et P_i à jour.

Pour la majorité des analyses, quand il n'y a pas de changement de note entre i et $i-1$, f_i et f_{i-1} sont de valeur proche. Cependant, selon le mode de jeu du musicien et les caractéristiques de l'instrument, la fréquence d'une même note peut varier de façon significative (quand il y a un vibrato par exemple).

Il est très important de noter qu'une note, en tant qu'unité mélodique, peut en réalité couvrir plusieurs notes du point de vue fréquentiel. Si un musicien module une même note, notre système ne doit pas envoyer d'information correspondant à une nouvelle note. Notre système doit avoir une mémoire du passé.

Soit ε un seuil paramétrable, si $|f_i - f_{i-1}| < \varepsilon$ nous pouvons considérer qu'il n'y a pas de nouvelle note mais qu'il y a eut modulation de la même note. Nous n'envoyons pas d'information de nouvelle note, i.e. nous avons toujours la même note Midi qu'à l'étape précédente. Seule la valeur du pitch bend est mise à jour. P_i correspondra alors à l'écart entre la valeur fréquentielle de la note Midi $Mi = M_{i-1}$ et la fréquence f_i.

Tant que P_i est inférieur à un certain seuil α, nous considérons que nous avons toujours la même note. Aucune nouvelle note Midi M n'est envoyée. Si P_i dépasse ce seuil, le *pitchtracker* considère que la note Midi a changé ; elle prend alors la valeur Midi correspondant à la fondamentale en cours. L'étendue du pitch bend est paramétrable.

5 Paramétrages

La taille de la FFT Ce paramètre est fonction de l'instrument à détecter. Plus la tessiture est aiguë, plus la taille de la FFT peut être petite. En effet, pour de grandes fréquences nous pouvons choisir de petite fenêtres d'analyse. De plus, nous avons vu en 2.1 que l'écart entre deux composantes du signal devait être supérieur à $\frac{R}{N}$. Plus la fréquence est élevée, plus l'écart entre deux harmoniques successifs est grand et plus la valeur de N peut être petite. Pour toutes les mesures, il faut choisir N le plus petit possible afin d'avoir la meilleure réponse en temps. La taille de la FFT est toujours une puissance de deux en raison des algorithmes de FFT optimisés pour des fenêtres de taille 2^n.

Un instrument tel que la flûte, ayant pour note la plus grave Do3 de fréquence 262 Hz, donne de bons résultats avec des fenêtres de 512 points soit 11 mls.

Étendu du pitch bend : α En fonction du type de détection désiré, nous pouvons choisir l'étendue du pitch bend. Une valeur faible limitera la possibilité de modulation mais renverra plus de notes qu'une valeur importante qui permettra une modulation de hauteur étendue.

Stabilité en fréquence entre deux mesure : ε ε est le seuil pour lequel deux fréquences successives sont considérées comme identiques. Cela permet de décider de la continuité d'une note modulée.

En d'autres termes, deux fréquences f_1 et f_2 sont identiques si et seulement si $|f_1 - f_2| < \varepsilon$.

Stabilité en temps Une note est considérée comme stable donc comme existante si elle se reproduit un certain nombre de fois. Ce paramètre permet de déterminer le nombre minimum de fréquences identiques et successives nécessaires pour déclencher une nouvelle note.

Accord

L'accord de l'instrument détermine la conversion fréquence/Midi et la valeur du pitch-bend. Nous devons donc paramétrer notre système en fonction de l'instrument.

Seuils du *noise gate* Le signal d'entrée passe par un *noise gate.*

Il y deux seuils : haut (entrée) et bas (sortie). Une note est déclenchée si elle passe le seuil haut et est éteinte si elle passe sous le seuil bas.

Dynamique Détermine la dynamique de la réponse en volume en fonction de l'amplitude de la note détectée.

6 Interface avec le système utilisateur

Notre système renvoie plusieurs éléments.

On : une note Midi quand une nouvelle note stable est détectée, 0 sinon. L'utilisateur doit traiter cette information quand elle est différente de 0 comme un NoteOn.

Off : une note Midi quand une note auparavant détectée ne l'est plus, 0 sinon. L'utilisateur doit traiter cette information quand elle est différente de 0 comme un NoteOff.

Vol : une valeur comprise entre 0 et 127, fonction du niveau de la note détectée. Cette information peut être soit la vélocité d'une note lors d'un NoteOn soit traitée comme un contrôleur de volume lorsqu'une note est en cours.

Bend : cette information doit être traitée par un contrôleur du type *pitchwheel.*

Intégration dans un système externe Notre détecteur est intégrable dans tout type de système utilisant le Midi. Plusieurs implémentations utilisant MidiShare ont été réalisées notamment sous Linux, Macinstosh et Windows. Les FFT sont calculées avec la *Fastest Fourier Transform in the West (FFTW)* du MIT. Ces implémentations fournissent une interface utilisateurs permettant d'ajuster les différents paramètres et gèrent les messages Midi.

Une bibliothèque C++ du *pitchtracker* a été réalisée et permet une réutilisation aisée. Le système extérieur se charge de l'acquisition des données et les fournit au *pitchtracker.* Le *pitchtracker* renvoie les informations sur la note que le système utilisateur doit transformer en Midi.

7 Conclusion

Nous avons réalisé un détecteur de hauteur de note ou *pitchtracker* tirant avantage d'une amélioration originale du vocodeur de phase. Une utilisation dans de nombreux systèmes est possible et un paramétrage souple permet de l'adapter à de nombreux instruments. De nombreuses améliorations restent encore possibles notamment sur le contrôle du volume. Un paramétrage automatique en fonction de l'instrument est à étudier.

Relativement peu de tests ont été réalisés (surtout des tests avec flûte). Des outils de tests systématiques seraient à mettre au point. Ils permettraient de mesurer la fiabilité du système en fonction des différents paramètres et faciliteraient les réglages automatiques.

Références

[1] M. Dessainte-Catherine et S. Marchand, "*High Precision Fourier Analyse of Sounds using Signal Derivatives*", SCRIME 1998, Bordeaux

[2] S. Marchand, "*Improving Spectral Analyse Precision with an Enhanced Phase using Signal Derivatives*", SCRIME 1998, Bordeaux

[3] M. S. Puckette, T. Apel, D.D. Zicarelli, "*Real-time audio analysis tools for Pd and MSP*", Proceedings of the ICMC 1998, p109-112, 1998

[4] Ö. Izmirli, S. Bilgen, "*Multiple Fundamental Tracking for Polyphonic Note Recognition*", Recherches et applications en informatique musicale, pp. 305-314, Hermes 1998, Paris

[5] A. M. Noll, "*Pitch determination of human speech by the harmonic product spectrum, the harmonic sum spectrum , and a maximum likelihood estimate*", Proceedings of the Symposium on Computer Processing in Communications, Vol. XIX, pp. 779-797, Polytechnic Press 1970, New York.

[6] E. Leipp, Acoustique et Musique, Masson 1984, Paris

[7] M.S. Puckette, "*Phase-locked Vocoder*",Proceedings of the 1995 IEEE ASSP Conference on Applications of Signal Processing to Audio and Acoustics,1995, New-York

[8] M. Dolson, "*The phase vocoder : a tutorial.*" Computer Music Journal, 10/4 : pp. 14-26,1986

Transmission d'événements musicaux en temps réel sur Internet

Dominique Fober Yann Orlarey Stephane Letz

Grame - Centre National de Création Musicale
9, rue du Garet BP 1185
69202 LYON CEDEX 01
Tél +33 (0)4 720 737 00 Fax +33 (0)4 720 737 01
[fober, orlarey, letz]@grame.fr

Résumé

Nous présentons un nouveau protocole s'appuyant sur UDP, permettant de transmettre des événements datés en temps réel et fournissant au récepteur, les moyens d'une restitution temporelle correcte. Ce protocole inclus des mécanismes permettant de compenser la latence du réseau et d'optimiser l'utilisation de la bande passante. Il prend également en compte les dérives d'horloges des différentes machines impliquées dans une transmission. Il est particulièrement adapté à la transmission d'événements musicaux tels que les messages MIDI.

1 INTRODUCTION

Les problèmes liés à la transmission en temps réel de données multimédia sur des réseaux ont été identifiés au début des années 90 en terme ressources nécessaires pour la communication temps réel [1], prenant en compte les délais de transmission, la fiabilité ainsi que les besoins en bande passante. Les flots de données multimédia sont généralement gourmants en bande passante, c'est pourquoi une attention particulière a été accordée au débit et des protocoles tels que Internet Stream Protocol (ST2) [2] ou Resource ReSerVation Protocol (RSVP) [3] [4] ont été développés pour offrir des services de transmission efficaces aux applications ayant des contraintes de qualité de service. Ces protocoles opèrent notamment grace à une politique de réservation de ressources tout au long des noeuds situés sur le chemin de distribution des données. Bien que cette politique n'affecte pas le routage en lui même, elle suppose la collaboration des routeurs afin de maintenir l'allocation de ces ressources durant toute la session. Le principe de la réservation de ressouces a également été pris en compte dans des couches de protocoles plus bas niveau tel que ATM (specifié en 1991 par CCITT I.361), et l'intégration de services temps réel dans une architecture de réseau IP-ATM a été envisagée en utilisant ST2 ou RSVP [5].

Elaboré en dehors des préoccupations de qualité de service, le protocole RTP (Transport Protocol for Real-Time Applications) [6] quant à lui, ne fournit aucune garantie concernant les services temps réel. Il opère cependant en parallèle d'un protocole de contrôle (RTCP) qui permet un contrôle de la transmission ajustable à de larges réseaux multicast, notamment par le biais d'envois réguliers de reports RTCP contenant des informations sur la qualité de la transmission. Comme RTP et RTCP ont été conçu de manière indépendante des couches de transport sous jacentes, ils peuvent constituer une solution tout à fait efficace en association avec des protocoles de réservation de ressource tel que ST2.

A condition que leur taille soit limitée, la transmission en temps réel d'événements diffère des flots de données audio ou vidéo, notamment parce qu'ils n'ont pas de besoin particulier en bande passante. La transmission de données au format MIDI peut être effectuée en beaucoup moins de temps que nécessaire pour les jouer : l'Invention à deux voix en do majeur de J.S. Bach (BWV 772) sous forme d'un fichier MIDI de 6646 octets, représentant environ 1 minute 30 de musique, peut être transmise en 2 secondes via UDP sur une liaison à 28.8 kbps. C'est pourquoi la transmission brute de fichier est suffisante pour la plupart des applications, éventuellement associée à des techniques de bufferisation dans le cas de fichiers importants,

pour permettre leur restitution avant la fin du transfert.

Cette technique n'est cependant pas utilisable quand les données sont générées en temps réel : dans ce cas, des conventions supplémentaires entre l'émetteur et le récepteur sont nécessaires. De même que pour les flots de données audio, elles devront prendre en compte les délais de transmission et la latence du réseau, mais d'une manière particulière puisqu'il faudra à réception, pouvoir restituer l'ordonnancement temporel des événements transmis.

Pour ce qui est de la bande passante et bien qu'elle ne pose problème, une attention particulière doit être donnée à l'optimisation des paquets, car des transmissions trop fréquentes d'événements pourraient alors encombrer le réseau : un plein débit MIDI par exemple, représente un paquet de 3 octets MIDI toutes les millisecondes. Dans le cas d'une transmission via UDP, le surcoût des couches de transport sous-jacentes représente plus de 91% du contenu du paquet, ce qui ne constitue pas une solution satisfaisante du point de vue efficacité.

Enfin, un des problèmes non traité par les travaux précédents est lié à la synchronisation du temps de l'émetteur et du récepteur. Dans les faits, nous ne nous intéresserons qu'à la différence de fréquence entre les horloges. En l'absence de mécanisme de correction, la dérive du temps pourra apparaitre comme une augmentation constante de la latence du réseau et à terme, empêcher une restitution correcte des événements transmis.

Il y a peu de travaux traitant de flots de données temps réel dans le contexte particulier d'événements datés. Young et Fujinaga ont présenté un travail appliqué à des master classes de piano via Internet [7]. Basé sur OTUDP (Open Transport UDP), un objet Max transportant des données via UDP, il est essentiellement axé sur la perte de paquets. La synchronisation du temps n'y est pas traitée. Un autre travail appliqué à la transmission temps réel de données MIDI sur Ethernet [8] inclus des techniques d'optimisation de la bande passante qui seront reprises dans le protocole présenté.

La suite de cet article est structurée comme suit : la section 2 présente les techniques utilisées pour compenser la latence du réseau et pour optimiser la transmission des paquets. La section 3 traite des mécanismes permettant de corriger les dérives du temps. La section 4 présente l'implémentation du protocole et des résultats expérimentaux. La section 5 conclut en soulignant les développements futurs de ce travail.

1.1 Terminologie

Quelques termes utilisés dans cet article sont définis comme suit :

restitution temporelle : le processus de restitution temporelle garantit que des événements ordonnés dans le temps seront restitués avec le même ordonnancement temporel du coté du récepteur.

la latence du transport : représente le temps de transport d'un niveau de protocole au même niveau de protocole sur la machine distante. Le niveau utilisé ici est celui du protocol proposé, ce qui signifie que la latence du transport inclus aussi bien la latence du réseau que celle des couches logicielles qui traitent les différentes couches de protocole.

le délai de restitution : soit d_a, la date d'un événement sur une machine A et d_b, la date de restitution de cet événement sur une machine B. Considérant que d_a et d_b sont exprimés dans une référence de temps commune à A et B, le délai de restitution D est définit comme d_b - d_a. Il inclut la latence du transport mais également le délai introduit pour la restitution temporelle.

le décalage d'horloges : représente la différence de temps entre deux horloges.

la dérive d'horloges : représente la différence de fréquence de deux horloges.

le décalage d'horloge apparent : c'est le décalage d'horloge augmenté de la latence de transport courante au moment de la mesure.

2 TRANSPORT ET RESTITUTION

Les mécanismes permettant de transmettre et de restituer les événements sont basés sur des travaux précédents concernant des flots de données MIDI temps réel sur Ethernet [8]. Ils reposent sur une *période de groupage* et sur une *latence maximale autorisée* pour effectuer une restitution temporelle correcte des événements transmis.

2.1 Période de groupage

La période de groupage a pour but de minimiser la fréquence des paquets transmis et d'optimiser le rapport entre la quantité de données transmise et le surcoût des couches de transport sous-jacentes. Elle représente la période pendant laquelle les événements seront accumulés avant d'être transmis sur le réseau. La valeur de la période de groupage fixe également la limite de fréquence de transmission des paquets. La figure 1 présente le nombre d'octets transmis par paquets en fonction de la fréquence des événements et pour des périodes de groupage allant de 10 à 200 ms. La fréquence des événements est exprimée en nombre d'événements par seconde. On considère que chaque événement est codé sur 5 octets. La figure 2 présente les rapports correspondants entre l'en-tête des protocoles sous-jacents et les quantités de données transmises. On considère que la taille de l'en-tête est de 44 octets.

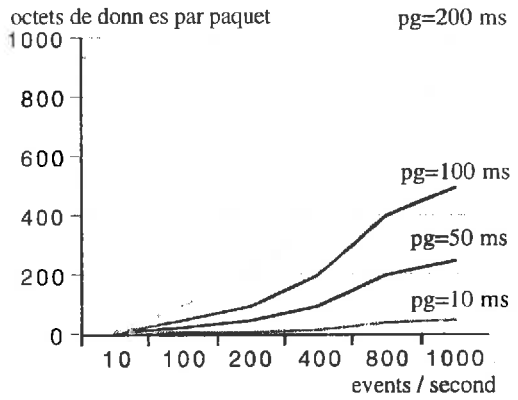


Figure 1: octets de données par paquet en fonction de la période de groupage et de la fréquence des événements.

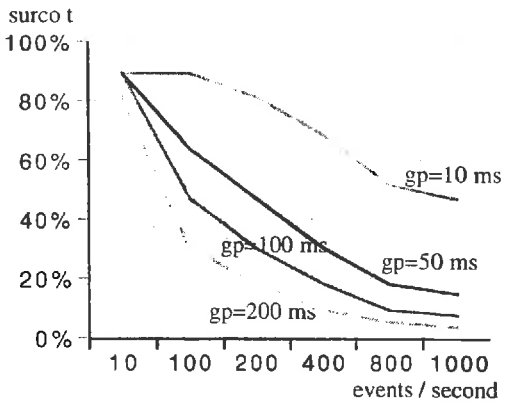


Figure 2: surcoût des protocoles sous-jacents en fonction de la période de groupage et de la fréquence des événements.

La période de groupage affecte le comportement de l'émetteur. Elle est incluse dans le délai de restitution. Ainsi que le montrent les travaux de Bolot [9], elle a également pour effet supplémentaire de minimiser la probabilité de perte de paquets et d'inversion due à des routes différentes de la source à la destination. Sa valeur sera fixée en fonction du contexte de transmission du réseau et de la fréquence des événements attendue. En particulier, cette valeur pourra être très basse si le protocole est destiné à opérer sur un réseau local (LAN).

2.2 Restitution temporelle

Comme la section suivante traite de la dérive d'horloges, nous allons considérer qu'elle est nulle pour expliquer le mécanisme de base de la restitution temporelle. Ce mécanisme repose sur une *latence maximale autorisée* afin de garantir un ordonnancement correct des événements transmis lors de leur restitution. Il opère de manière similaire aux techniques de bufferisation : la restitution des événements est différée dans le temps en fonction de la latence courante, ce qui revient à accumuler ces événements durant une période égale à la latence maximale autorisée.

Le paramètre de latence maximale autorisée affecte le comprtement du récepteur et sa valeur est également incluse dans le délai de restitution.

2.2.1 Evaluation de la variation de la latence.

En pratique, nous ne nous intéresserons uniquement à la variation de la latence du transport. Dans un premier temps, le décalage d'horloge apparent est mesuré à l'initialisation du protocole. Chaque paquet transmis comporte une date d'émission: soit A_n la date d'émission du nième paquet transmis par une machine A et B_n, la date de réception de ce paquet sur une machine B ; le décalage d'horloge apparent vu par la machine B est alors :

$$\Theta = B_o - A_o \qquad (1)$$

Il inclut alors la latence du transport.
Pour toute transmission ultérieure, la variation de la latence de A à B est évaluée comme suit :

$$\delta_n = B_n - A_n - \Theta \qquad (2)$$

2.2.2 Date des événements

Chaque événement transmis est daté par l'émetteur sous forme d'offset par rapport à la date du paquet qui le transporte. Le récepteur ajoute cet offset à la date de réception du paquet pour produire la *date locale de l'événement*. Soit B_n, la date de réception du nième paquet et o_i, l'offset de l'événement i relativement au paquet . La date locale de l'événement e_n^i est alors :

$$e_n^i = B_n + o_i \qquad (3)$$

Pour compenser la latence du transport, la date de restitution r_n^i de l'événement e_n^i est alors calculée comme suit :

$$r_n^i = e_n^i + L_{max} - \delta_n \qquad (4)$$

où L_{max} est la latence maximale autorisée.
A partir de (1, 2, 3), r_n^i peut être exprimé également par :

$$r_n^i = B_o + A_n - A_o + L_{max} + o_i \qquad (5)$$

ce qui est indépendant de la variation de la latence δ_n.
Cependant, pour que la restitution temporelle soit correcte, il faut s'assurer que la date de resitution r_n^i est supérieure ou égale à la date de réception du paquet B_n. Dérivé de (2), cela signifie que

$$B_o + A_n - A_o + L_{max} + o_i \geq A_n + \Theta + \delta_n \qquad (6)$$

et déduit de (1), c'est équivalent à :

$$L_{max} + o_i \geq \delta_n \qquad (7)$$

Considérant que o_i est nul pour le premier événement d'un paquet, cela signifie que la variation de la latence ne doit pas dépasser la latence maximale autorisée pour que la restitution temporelle soit correcte.

3 DERIVE D'HORLOGES

La dérive d'horloges peut être vue du coté récepteur, comme une augmentation constante de la latence du transport : soit R, le rapport de fréquence des horloges de deux machines A et B. Si l'on considère que la latence du transport est nulle, alors en déduction de (2) nous avons :

$$B_n = A_n + \Theta \qquad (8)$$

cependant, et en fonction de la dérive d'horloges, la date de réception du nième paquet sur la machine B peut être exprimée comme :

$$B_n = \frac{(A_n + \Theta)}{R} \qquad (9)$$

ce qui signifie que la variation de la latence vue par la machine B est alors :

$$\delta_n = (A_n + \Theta)\frac{1-R}{R} \qquad (10)$$

Si le rapport de fréquence R entre les horloges A et B est égal à 1, alors la variation de la latence δ_n est effectivement nulle, mais si R est plus petit que 1 (ie l'horloge B est plus rapide que l'horloge A), alors δ_n augmente avec A_n et atteindra plus ou moins vite la latence maximale autorisée, empêchant ainsi une restitution temporelle correcte des événements transmis.

3.1 Techniques relatives

La synchronisation d'horloges a fait l'objet de nombreux travaux. Parmis eux, le protocole NTP (Network Time Protocol) [10] permet la synchronisation d'horloges réparties sur Internet. Son utilisation en parallèle du protocole n'est pas envisagée en raison de certaines limitations : en particulier, la précision donnée par NTP est directement dépendante du temps mis pour l'obtenir et plusieurs heures et des douzaine de mesures sont nécessaires pour maintenir le temps avec une précision de l'ordre de la milliseconde.

La synchronisation d'horloge fait également partie des problématique de base des systèmes distribués. Dans ce cadre, un certain nombre d'algorithmes de synchronisation d'horloge ont été développés [11, 12, 13] pour permettre à des processus physiquement dispersés d'acquérir une notion du temps commune par le biais d'horloges physiques locales et d'échanges de messages via un réseau. Bien qu'en moins grand nombre, certaines recherches s'appliquent plus spécifiquement la synchronisation de fréquence des horloges [14, 15]. Tous ces travaux s'appliquent généralement à des systèmes répartis comportant de nombreux noeuds. Une approche commune consiste à utiliser des algorithmes de synchronisation tolérants aux fautes et notamment par le biais d'algorithmes de convergence interactive [16, 17] tels que le *sliding window algorithme* (SWA), le *fault-tolerant midpoint algorithme* (FTMA), l'*adaptative exponential fault-tolerant midpoint algorithme* (AEFTMA) ou le *multistep interactive convergence algorithme* (m-ICV).

La solution que nous avons adopté pour la détection de la dérive d'horloges est basée sur ces algorithmes de convergence.

3.2 Algorithme de détection

Comparé aux travaux précédents sur la synchronisation d'horloge et de fréquence, notre contexte d'opération constitue un cas particulier de plusieurs points de vue :

- seulement deux noeuds sont impliqués dans le processus de détection de la dérive d'horloges,
- il n'est pas nécessaire d'obtenir un consensus sur une base temporelle commune: chaque noeud peut estimer la dérive de son horloge de manière indépendante,
- comme la variation de la latence est utilisée pour détecter la dérive d'horloge, nous ne pouvons pas considérer certaines de ces valeurs comme étant fautives.

3.2.1 Filtrage des pics de latence

Nous avons fait un certain nombre de mesures de variation de latence en utilisant des routes de réseau différentes. Il apparait que cette variation est contenue globalement dans un éventail borné avec des pics plus ou moins fréquents. La largeur de l'éventail ainsi que l'amplitude des pics peut varier de manière importante en fonction des fournisseurs d'accès à Internet et des routes empruntées. La figure 3 montre les résultats de mesures continues effectuées via une connection modem à 42,6 kbps et un fournisseur d'accès libre (sans abonnement), à une heure réputée comme étant une heure de forte fréquentation. La mesure a été faite en envoyant des paquets datés toutes les 200 millisecondes. La légère pente de l'augmentation constante de

latence correspond à la dérive d'horloge mesurée entre les 2 machines.
La figure 4 donne la route correspondante sur le réseau.

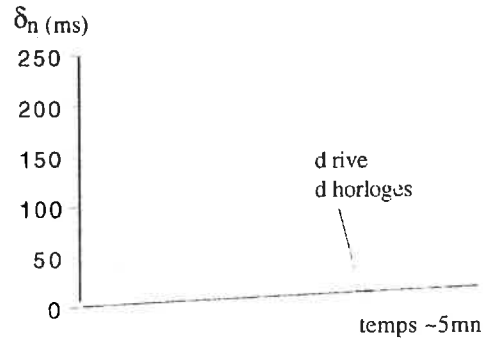


Figure 3: mesure de la variation de la latence pendant 5 mn

1	lyon2-2-58-254.dial.proxad.net
2	paris11-2-a1.routers.proxad.net
3	telehouse-6.routers.proxad.net
4	teleglobe.net
5	if-0-0.core1.paris.teleglobe.net
6	No host name found.
7	if-1-7.core1.newyork.teleglobe.net
8	ix-1-8.core1.newyork.teleglobe.net
9	jfk-core-02.inet.qwest.net
10	jfk-core-01.inet.qwest.net
11	chi-core-01.inet.qwest.net
12	chi-edge-01.inet.qwest.net
13	ar-chicago-cern.ch
14	cernh9-pos500.cern.ch
15	in2p3-fddi.in2p3.fr
16	lyon-inter.in2p3.fr
17	lyon-tif.in2p3.fr
18	grame-cisco.in2p3.fr
19	bach.grame.fr

Figure 4: routage correspondant à la figure 3

La solution proposée consiste à filtrer les pics de latence et à calculer le point de convergence des valeurs restantes. Nous avons appliqué un algorithme dérivé de FTMA à la variation de la latence pour obtenir ce que nous nommerons un *profil de dérive d'horloges*.

Appliqué à la synchronisation d'horloges, le *fault-tolerant midpoint algorithme* (FTMA) repose sur l'hypothèse qu'au plus k horloges sont fautives lors de chaque opération de resynchronisation. Dans ce cas, il faut que le système comporte au moins $n = 3k + 1$ noeuds pour tolérer des fautes k Byzantines [21]. Pour déterminer la correction d'horloge, FTMA élimine les k déviations d'horloges supérieures et inférieures et calcule ensuite la moyenne arithmétique des déviations extrêmes restantes [16].

Notre algorithme, que nous nommerons *peak-tolerant midpoint algorithm* (PTMA) opère de manière similaire avec cependant les différences suivantes :

- le vecteur ordonné de déviations d'horloge, constitué par FTMA à partir des valeurs collectées auprès des n noeuds du système, est transformé en un vecteur ordonné de variations de latence collecté dans le temps,
- le nombre de valeurs éliminées n'est pas limité aux tolérances k Byzantine,
- la moyenne arithmétique est calculée en utilisant l'ensemble des valeurs résiduelles (et non plus les seules extrêmes).

Soit w, la taille de la fenêtre temporelle de collection de la variation de la variation de latence et k, le nombre de valeurs éliminées par fenêtre. A une date t, le vecteur ordonné de la variation de latence est

$$\overline{\Delta_t} = [\delta_{i-w} \ldots \delta_i \ldots \delta_t], \delta_i \leq \delta_{i+1}$$

et le point de convergence de cette variation est calculé comme suit :

$$LV_t = \frac{\sum_{i=t-w+1/2}^{i=t-1/2} \delta i}{w-k} \qquad (11)$$

De manière intuitive, l'opération effectuée par l'algorithme peut être vue comme une sélection des variations de la latence (il élimine les variations les plus fortes et les plus faibles), mais également comme une sélection dans l'historique des variations étant donné qu'il peut ne retenir qu'un petit sous-ensemble discontinu des valeurs d'une fenêtre temporelle. La figure 5 illustre le résultat de cet algorithme appliqué aux mesures présentées ci-dessus. La ligne de

correction d'horloge représente la sortie de l'algorithme PTMA.

Pour cette application, la taille de la fenêtre temporelle est de 50 et 10 valeurs sont retenues par fenêtre. Les mesures ont été effectuées toutes les 200 ms, cela signifie qu'un sous-ensemble représentant 500 mls de variations est retenu pour 10 secondes de mesures.

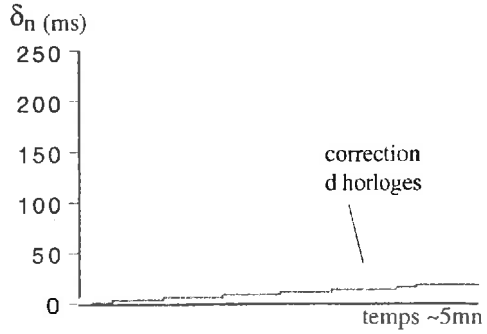


Figure 5: correction en sortie de PTMA

Notez que ce profil de dérive d'horloges peut également inclure des changements globals à long terme dans la latence du transport, ce qui satisfait également à nos besoins.

3.2.2 Lissage du profil de dérive d'horloges

Dans l'exemple précédent, les conditions de transmissions étaient plutôt bonnes comparées à d'autres mesures (effectuées via d'autres fournisseurs d'accès à Internet et donc d'autres routes sur le réseau) qui laissent apparaitre des conditions de transmission beaucoup plus chaotiques, avec des pics de latence dépassant les 3 secondes et un éventail constant pouvant atteindre 200 ms sur des périodes dépassant la largeur de la fenêtre temporelle. Nous avons donc étendu l'algorithme PTMA, que nous nommerons alors *exponential peak-tolerant midpoint average algorithm* (EPTMA) en appliquant un lissage exponentiel aux valeurs en sortie de PTMA.

Soit LV_t, la valeur du profil de dérive d'horloges à une date t, EPTMA calcule la valeur lissée corespondante comme:

$$LV^t_{EPTMA} = \alpha.LV_t + (1-\alpha).LV^{t-1}_{EPTMA} \qquad (12)$$

où le facteur de pondération α est tel que $0 \leq \alpha \leq 1$.

Une petite valeur pour α donne plus de poids aux valeurs passées et minimise les effets de variations brutales tout en les prolongeant dans le temps. Supposons que la latence a une valeur constante L et qu'à une date t, cette valeur passe sans transition à une valeur L'. Soit R, le rapport entre L et L'. A une date $t+\theta$, le rapport entre la valeur lissée et L' est exprimé par :

$$1 - \frac{(1-\alpha)^{\theta+1}(R-1)}{R} \qquad (13)$$

La figure 6 illustre la réponse du filtrage dans le temps pour $R = 1.3$ et différentes valeurs du facteur de pondération α.

Il apparait que même avec un facteur très petit, l'erreur de correction peut être considérée comme négligeable (au regard de nos besoins) après une minute.

La figure 7 illustre un profil de dérive d'horloges lissé, constitué à partir des pires conditions de transport mesurées. Le facteur de pondération α correspondant est de 0,01. La dernière partie du diagramme montre des distortions importantes dans le profil de dérive des horloges. Bien que ces distortions soient partiellement reportées sur le profil lissé, sa forme globale reste correcte et la distortion introduite dans la restitution temporelle est grandement minimisée. La résolution du profil de dérive lissé est la milliseconde, ce qui explique les paliers de la courbe.

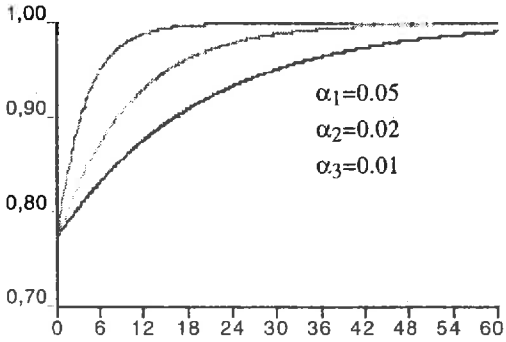


Figure 6: réponse dans le temps à une variation de la latence.

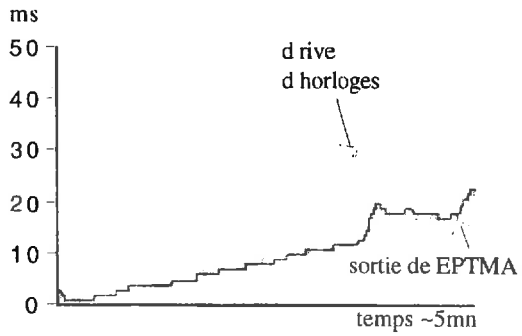


Figure 7: sortie de EPTMA comparé à PTMA

3.2.3 Correction de la dérive d'horloges

Ainsi que décrit au paragraphe 2.2, la restitution temporelle des événements est basée sur le décalage d'horloges apparent mesuré à l'initialisation du protocole. Une machine B connectée à une machine A maintient 2 dates : la date d'initialisation locale B_O et la date apparente correspondante de la machine distante A_O. Pour compenser la dérive d'horloges, la date r_n^i de restitution d'un événement e_n^i exprimée en (5) peut maintenant être corrigée de la manière suivante :

$$r_n^i = B_o + LV_{EPTMA}^n + A_n - A_o + L_{max} + o_i \quad (14)$$

où LV_{EPTMA}^n représente la valeur lissée du profil de dérive calculée à la date n. C'est également équivalent à l'ajout de tout changement dans les valeurs produites par EPTMA à la date locale d'initialisation B_O de manière à maintenir un décalage d'horloge apparent constant dans la référence temporelle de A. Cette technique pour compenser la dérive d'horloges permet d'éviter le calcul d'un rapport de fréquences ainsi que les conversions de dates d'une référence temporelle à une autre. De plus, le mécanisme de compensation peut alors opérer de manière indépendante du mécanisme de restitution temporelle, ce qui facilite grandement l'implémentation du protocole.

4 IMPLEMENTATION DU PROTOCOLE

L'implémentation actuelle du protocole transmet des événements MidiShare [18, 19, 20] qui sont des événements de haut niveau, structurés et datés avec une résolution de la milliseconde. Leur typologie inclut des événements au format MIDI ainsi qu'au format MIDIFILE. La couche de transport sous-jacente utilisée est UDP (User Datagram Protocol). Deux variations sur le protocole de base sont présentées, conçues pour optimiser les implémentations LAN et WAN.

4.1 Format des paquets de base

Tous les messages ont un en-tête commun illustré par la figure 8. Les différents champs de l'en-tête ont la signification suivante:

- ID: identificateur de protocole sur 16 bits, permet d'éliminer les paquets étrangers au protocole.
- Version: numéro de version du protocole sur 8 bits, actuellement *un* (1).
- Type: identificateur de message sur 8 bits, permet de différencier les types de paquets.

Le protocole s'appuie sur 2 paquets de base : les *paquets d'événements*, qui transportent des événements datés, et les *paquets d'identification*, dont le but est à la fois de permettre l'identification des machines sur le réseau et d'assurer la continuité de la détection de dérive

d'horloges quand il n'y a pas de paquets d'événements à transmettre. Les deux types de paquets sont datés avec une résolution à la milliseconde.

4.1.1 Paquets d'événements

Le format des paquets d'événements est illustré en figure 9.

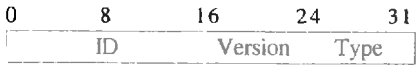


Figure 8: en-tête commun à tous les messages

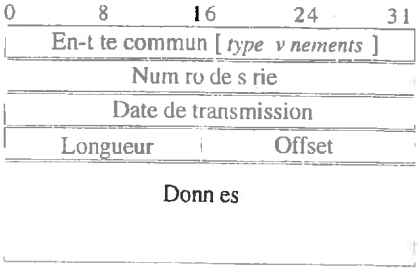


Figure 9: format des paquets d'événements

Les différents champs sont les suivants :

- En-tête commun : ainsi que décrit ci-dessus
- Numéro de série : un numéro de série unique sur 32 bits, incrémenté pour chaque paquet transmis et permettant de détecter les pertes de paquets.
- Date de transmission : une date en millisecondes sur 32 bits, exprimée dans la référence temporelle de l'émetteur.
- Longueur : le nombre de données transmises codé sur 16 bits, il représente la longueur du champ *Données*.
- Offset: une valeur sur 16 bits qui représente l'offset de début du premier événement dans le champs *Données*. La plupart du temps, sa valeur sera de 0, elle pourra être supérieure dans le cas d'un gros événement ne tenant pas dans un seul paquet. Il permet d'assurer la continuité de la lecture des données dans le cas de pertes de paquets.
- Données : un champ de longueur variable contenant les données correspondant aux événements transmis. La seule contrainte concernant le format de ces données concerne la date des événements : ainsi qu'indiqué précédemment, chaque événement doit être daté sous forme d'offset par rapport à la date du paquet qui le transporte. Cette date est actuellement expimée avec une résolution d'une milliseconde et 2 octets sont suffisants pour couvrir l'étendue des offsets possibles.

La définition actuelle du protocole ne fournit pas de mécanisme de récupération en cas de perte de paquet.

4.1.2 Paquets d'identification

Le format des paquets d'identification est illustré en figure 10. De même que pour les paquets d'événements, la date de transmission est un champ de 32 bits contenant une date exprimée en millisecondes dans le temps de référence de l'émetteur. Le *Nom* est un champ de longueur variable contenant le nom symbolique de l'émetteur.

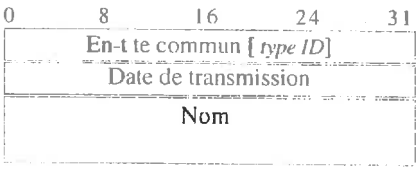


Figure 10: format d'un paquet d'identification

L'utilisation actuelle des paquets d'identification diffère légèrement dans les implémentations LAN et WAN. Toutefois, leur fonction commune est d'assurer la continuité de la détection de dérive d'horloges.

4.2 Opérations spécifiques LAN et WAN

Les implémentations pour LAN et WAN diffèrent dans leur manière d'établir une connexion et de maintenir la continuité de la détection de dérive d'horloges. Ils utilisent également des types de paquets supplémentaires pour la gestion des connexions. Enfin, pour l'implémentation LAN, les paramètres de période de groupage et de latence maximale autorisée sont définis par défaut de manière à minimiser le délai de restitution.

4.2.1 Processus de connexion

L'implémentation LAN inclus un processus de connexion automatique : à l'initialisation du protocole, un paquet d'identification est envoyé sur l'adresse de broadcast du réseau pour être émis ensuite toutes les 200 ms. Toute machine sur le réseau recevant un paquet d'identification est censée allouer les ressources nécessaires pour gérer la connexion correspondante (si elle n'existe pas encore). Quand ce paquet disparait, la machine correspondante est considérée comme inaccessible et la connexion doit être détruite.
L'implémentation WAN utilise des transactions TCP pour la gestion des connexions.

4.2.2 Continuité de la détection de dérive d'horloges

Pour maintenir la détection de dérive d'horloges dans le temps, une machine doit recevoir des paquets datés à une fréquence suffisante. L'émission régulière de paquets d'identification dans l'implémentation LAN assure cette continuité dans le temps.
L'implémentation WAN quant à elle opère de manière à minimiser la transmission de paquets : les paquets d'identification ne sont envoyés que s'il n'y a pas de paquet d'événements à transmettre.

4.2.3 Paquets optionnels

Deux types de paquets supplémentaires sont définis pour une meilleure gestion des connexions

- un paquet *Connection Refusée* : utilisé uniquement dans l'implémentation WAN et via TCP, il contient la raison du refus.
- un paquet *Bye* : envoyé lors de la fermeture d'une connexion, il contient des informations statistiques sur le déroulement de la session parmi lesquelles : le nombre de paquets perdus, la valeur du paramètre de latence maximale et le nombre de dépassements de latence. Pour l'implémentation LAN, c'est le seul moyen de fermer proprement une session.

4.3 Expérimentation

4.3.1 Paramétrage du protocole

Les expérimentations du protocole faites sur LAN et sur WAN ont utilisé les paramétrages suivants :

paramètre	*valeur LAN*	*valeur WAN*
période de groupage	10 ms	200 ms
latence maximale	10 ms	1500 ms

Les délais de restitution correspondants étaient donc de 20 ms sur LAN et au moins de 1700 ms sur WAN.
L'algorithme EPTMA faisait également usage de paramétrages différents :

paramètre	*valeur LAN*	*valeur WAN*
fenêtre temporelle	10	50
valeurs retenues	6	10
facteur de pondération	0.1	0.01

4.3.2 Mise en place d'un serveur Internet

Afin de valider le protocole, nous avons mis en place un serveur qui diffuse un flot d'événements MIDI en continu à partir de l'adresse "radio-hd.grame.fr". Basé sur le principe d'une radio, ce serveur génère son contenu musical en temps réel, à partir de n'importe quel fichier présent sur le disque dur du serveur, qui est hébergé actuellement sur une machine Linux. Des clients spécifiques sont disponibles pour Macinosh et Linux à l'adresse suivante: http://www.grame.fr/radio-hd.

5 CONCLUSION

Nous avons proposé un protocole simple pour transmettre en temps réel sur Internet et restituer des événements ordonnés dans le temps. La solution proposée présente de nombreux avantages : absence de transaction pour pouvoir opérer, indépendance des machines connectées, évaluation asymétrique de la dérive d'horloges, facilité et légèreté de l'implémentation. Toutefois, des améliorations possibles restent à explorer : en particulier concernant la dérive d'horloges, des algorithmes adaptatifs basés sur PTMA et EPTMA pourraient produire de meilleurs résultats. Le support d'adresses mulitcast permettra également d'optimiser le traffic pour des utilisations de type client / serveur.
Le code source de l'implémentation actuelle est disponible sous license LGPL (Library General Public License) à l'adresse suivante: http://www.grame.fr/MidiShare/SCPP/.

REMERCIEMENTS

Cette recherche a été conduite avec la participation de la Société Mil Productions (Villefranche/Saone - France). Nous tenons à la remercier pour sa contribution.

REFERENCES

[1] D.Ferrari. Client requirements for real-time communication services. *RFC 1193 (Status: informational)* Nov-01-1990.

[2] L. Delgrossi, L. Berger. Internet Stream Protocol Version 2 (ST2) Protocol Specification - Version ST2+.*RFC 1819 (Status: experimental)* August 1995.

[3] R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, S. Jamin. Resource ReSerVation Protocol (RSVP) *RFC 2205 (Status: proposed standard)* September 1997.

[4] S. Herzog. RSVP Extensions for Policy Control. *RFC 2750 (Status: proposed standard)* January 2000.

[5] M. Borden, E. Crawley, B. Davie, S. Batsell. Integration of Real-time Services in an IP-ATM Network Architecture. RFC 1821 August 1995.

[6] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. Audio-Video Transport Working Group. *RFC 1889 (Status: informational)* January 1996.

[7] J.P. Young, I. Fujinaga. Piano master classes via the Internet. *Proceedings of the International Computer Music Conference 1999.* ICMA San Francisco, 1999, pp.135-137

[8] M. Wright. Implementation and performances issues with OpenSound Control. *Proceedings of the International Computer Music Conference 1998*, ICMA San Francisco, 1998, pp. 224-227

[8] D. Fober. Real-Time Midi data flow on Ethernet and the software architecture of MidiShare - *Proceedings of the International Computer Music Conference 1994*, ICMA San Francisco, 1994, pp. 447-450

[9] J.C. Bolot. End-to-end packet delay and loss behavior in the internet. *Conference proceedings on Communications architectures, protocols and applications.* ACM, 1993, pp.289-298

[10]D.L. Mills. Network Time Protocol (Version 3) Specification, Implementation. *RFC 1305 (Status: draft standard)* March 1992.

[11]T.K. Srikanth, S. Toueg. Optimal Clock Synchronization. *Journal of the ACM*, vol. 34, pp. 626–645, July 1987.

[12]B. Patt-Shamir, S. Rajsbaum. A theory of clock synchronization (extended abstract). *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, 1994, pp. 810 - 819

[13]R. Ostrovsky, B. Patt-Shamir. Optimal and efficient clock synchronization under drifting clocks. *Proceedings of the eighteenth annual ACM symposium on Principles of distributed computing*, 1999, pp. 3 - 12

[14]K. Schossmaier. An interval-based framework for clock rate synchronization. *Proceedings of the sixteenth annual ACM symposium on Principles of distributed computing*, 1997, pp. 169 - 178

[15]K. Schossmaier, B. Weiss. An Algorithm for Fault-Tolerant Clock State and Rate Synchronization. *Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems*, 1998

[16]M.M. de Azevedo and D.M. Blough, "Fault-Tolerant Clock Synchronization for Distributed Systems with High Message Delay Variation," *1994 IEEE Workshop Fault-Tolerant Parallel and Distributed Systems. (Appears in Fault-Tolerant Parallel and Distributed Systems, D. Pradhan and D. Avresky, eds., pp. 268–277, IEEE CS Press, 1995.*) Computing, vol. 27, pp. 1–14, May 1995.

[17]M.M. de Azevedo, D.M. Blough. Multistep Interactive Convergence: An Efficient Approach to the Fault-Tolerant Clock Synchronization of Large Multicomputers. *IEEE Transactions on Parallel and Distributed Systems 9(12)*, 1998, pp. 1195-1212

[18] Y. Orlarey, H. Lequay. MidiShare : a Real Time multi-tasks software module for Midi applications - *Proceedings of the International Computer Music Conference 1989*, ICMA San Francisco, 1989, pp.234-237

[19] D.Fober, Y. Orlarey, S. Letz. Recent developments of MidiShare - *Proceedings of the International Computer Music Conference 1996,* ICMA San Francisco, 1996, pp.40-42

[20] D.Fober, Y. Orlarey, S. Letz. MidiShare joins the Open Source Softwares - *Proceedings of the International Computer Music Conference 1999* ICMA San Francisco, 1999, pp.311-313

[21]L. Lamport, R. Shostak, M. Pease. The Byzantine Generals Problem, *ACM Transactions on Programming Languages and Systems*, Vol. 4, No. 3, July 1982, pp.382-401.

Earle Brown's "25 pianos": a web interactive implementation

Guigue, Didier [1]
Gomes de Andrade, Fábio [2]
Departamento de Música, Universidade Federal da Paraíba
João Pessoa (Paraíba), Brazil
dguigue@openline.com.br, fgandrade@ig.com.br

Summary

Earle Brown's "25 pages for 1 to 25 pianos" is a typical "open form" composition, which let a range of variables up to the performer's decision. We explain why it represents a good case study for a web interactive implementation, on a musical as well as on a computing point of view. Then we describe how this implementation is being done in Java, discuss problems, solutions and on-going developments [3].

Keywords: web interactive music implementation with Java; open form music.

Earle Brown's "open form" music

Among the members of the so-called "New York Group" (besides Morton Feldman, Christian Wolff and with John Cage as the senior member), Earle Brown (b. 1926) is, in the middle of the 50's, one of the most engaged composers in "open form" musical experiments. A radical break, his seminal graphical score "December 1952" appears to be « the first serious invitation to the classical musician to improvise rather than 'read' in the conventional sense » (Ryan 1999). He was strongly influenced by the painter Jackson Pollock and the sculptor Alexander Calder. « The experience of Alexander Calder's mobiles, long before Brown's meeting with Cage, had implanted in his mind the idea of mobility, and also the relationship of temporal and spatial concepts in music which had encouraged his experimentations » (*Ibid.*).
These concepts are crucial for the piece "25 pages for 1 to 25 pianos" (1953), where « he develops more consistently a notion of open form rather than the 'open content' of the graphic pieces » (*Ibid.*). « A temporal order can be pre-established by the performer, obtained from the composer, or arrived at spontaneously by the performer(s) » (Delaigue 1989). Brown attempts to find a path which embraces both extreme variability while maintaining an identity for the piece. « There must be a fixed (even flexible) sound content, to establish the character of the work, in order to be called 'open' or

---

[1] Doctor in 20th Century Music & Musicology by the EHESS/IRCAM (France). Professor at the Music Department of the Universidade Federal da Paraíba, João Pessoa (Brazil). Coordinator of the GMT — *Grupo de Pesquisas em Música, Musicologia e Tecnologia Aplicada*. Researcher and Consultor at the CNPQ (Brazilian Council for Research), in the domain of technology and computers applied to music.
[2] Graduating student at the Computers Department of the Universidade Federal da Paraíba. He develops this one-year Earle Brown's project under the orientation of Didier Guigue, with a grant from the Cientifical Research Initiation brazilian program.
[3] The authors thank Ernesto Trajano for his advice and help in programming.

'available' form. » (Brown, in Nyman 1999).

The "25 Pages..." variables

"25 Pages for 1 to 25 pianos" is a composition where a number of variables are up to the performer (s), in such a way the piece, as a Calder's mobile, will look different each time it is played, although its sonic identity always remains strongly present.
The musical notation is a blend of traditional elements — the two staves of the piano system, the horizontal reading from left to right, the proportional time representation — with less usual graphic symbols for pitches, durations and articulation. These symbols are very carefully explained in the composer's foreword, and the graduation of intensities and articulations is extremely subtle.

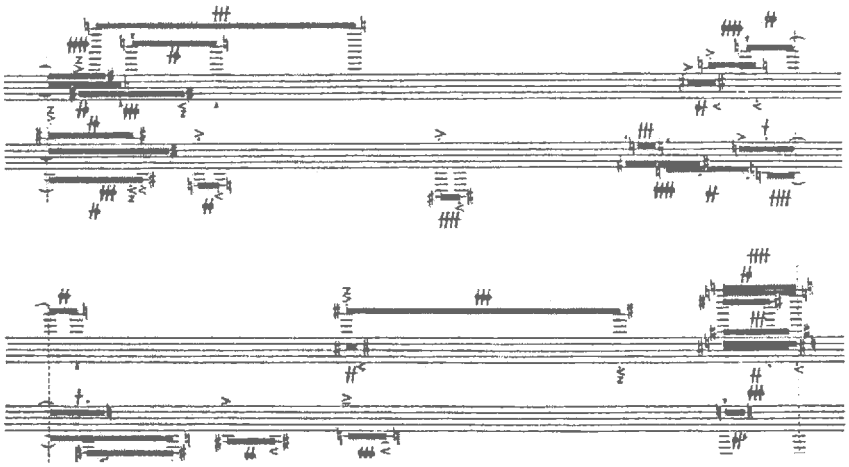


Fig. 1: An excerpt of the score: page 1, 3d & 4th systems.
© 1975 Universal Edition (Canada) Ltd, Toronto.

The variables of the piece are the following:
1. The 25 pages can be played in any order;
2. The pages can be read upside down;
3. The two line systems can be read in treble or bass clefs;
4. The total duration of the piece can range from 8 mn 20 s to 25 mn (not compulsory), for the player can choose different time resolutions;
5. The piece can be performed by 1 to 25 pianists.

Such "open form" pieces became common during that turning-point 20th century period. It is not the place here to discuss the historical context nor the conceptual background of this aesthetical option, which sounds nowadays somewhat obsolete, although it gave rise to other beautiful masterpieces such as, in the case of the piano repertoire, Boucourechliev's "Archipels", Boulez' "Third Sonata", Cage's "Chance Music", and some "Klavierstücken" by Stockhausen.

A web interactive simulation of "25 Pages..."

The Brown's composition represents a very interesting study case for a web interactive simulation:

- Its "full" version, with up to 25 pianists, has very few chances to be some day effectively performed on stage, for obvious reasons — what theater, what stage, what producer can afford 25 grand pianos with 25 good professional players at the same time? [4] Nevertheless, it suggests a very exciting and unique musical, sonic and spatial experience. Thus, there is no doubt that offering a simulation of this experience is of great musical and historical interest.
- Depending on decisions the performer (s) make for some variables, some moments of the score may turn impossible to sound or to be played: tones may fall outside the piano's range, some spreads may become excessively large for human hand, some unplayable unisons may appear, and so on.
- In some particularly dense moments, there is far more very subtle and almost microcospical indications of dynamics and attacks — generally applied independently to each tone — than the performer can securely play [5].
- Although, in seek of realism, these shortcomings could be introduced in a simulation computer program, they can also be bypassed to obtain an absolute version of the piece where all the variables effectively and systematically affect the final result — remembering that the MIDI protocol, for instance, let simulate a virtual piano with no less than 10 octaves and a performer able to control up to 128 loudness steps.
- Unlike Stockhausen's "Klavierstück n. 11" or Boulez' "Third Sonata", there is no need to be a musical expert to understand the variable rules and control a performance of "25 pages...". This is a very important point in dealing with an "all-audiences" web production.
- Another positive aspect is that the entry points of the interactivity — number of pianos and any of the other performance variables — are very clearly audible to any non prepared subject. This means instant gratification.

These are the reasons why we choose this piece to develop a web environment where the visitor, which acts as a kind of "maestro", "artistic director" or "composer's assistent", chooses and prepares his/her own version of "25 pages...", and immediately listens to it.

We introduce another variable Brown's does not formalize, but which is implicit for any performance with more than a single piano: the way the pianos are placed on a virtual stage. In a real performance in a concert hall, and assumed that there is no electronic balance nor equalization, each piano will reach the audience from a different point in the

[4] Till the present date, we have failed in finding some evidence of live or recorded performances of this piece with more than one single piano.

[5] The composer is fully acquainted with this problem: « There is clearly an excess of detailed information given as to the loudness, attack condition, duration and juxtaposition... excess, in the sense that all of the indications cannot be fulfilled in the more "dense" complexes » (Brown 1974). He suggests some adaptations to apply to the score in such moments.

space and with a different global loudness and timbre. Thus it is coherent with the composer's project the user can access to the performance stage and choose from where each piano will be heard.

The implementation

Audio vs MIDI solutions

When implementing a computer and/or web music application, the first question concerns the choice of the format. Will the "25 pages" software read, process and send forward MIDI or audio data? Assuming there is no need to describe here their respective properties, let us simply check the pros and contras of each format for each of the project's main musical constraints.

- *Artistic performance of the 25 pages, for later digitalization.* A profissional pianist recording the pages on an acoustic grand piano is obviously the most natural solution. The recordings would then be digitalized as audio files. Using a fully weighted digital piano, perfectly simulating the behaviour of a grand piano keyboard, the MIDI recording may also be an artistic satisfying solution, from the pianist point of view (but see next topic). If needed, a very precise transcription of the subtle dynamics and attacks directions of the written score can be strictly reached, by means of a computer-assisted edition of the recorded MIDI files, a very awkward task, if not impossible, with audio data.
- *Quality of the Piano sound.* As known, MIDI does not carry any sonic data. Thus, the quality of a MIDI virtual experience of "25 pages" depends exclusively on the quality of the MIDI piano device the end-user have connected to his/her computer. Æsthetically speaking, this is a serious matter of consideration, as Earle Brown's music is essentially based on sonic acoustic qualities, rather than on abstract pitch structures.
- *Storage of the 25 pages; size of files; transfer rates on the web.* Compared to the very large size an audio file of a single page of this music will must have, even with some digital compression [6], the tiny size of a standard MIDI file is undoubtly the best choice. Small size means fast transfer and small storage needs.
- *Accurate processing of pitches and durations, in order to execute in real time the user's defined " reversions" [7], transpositions and time unit variables.* It is a well known fact that computer's "musical" understanding of an audio file is far from a trivial problem, with no satisfactory solution for the while (Leão *et al.* 2001). There is no way to ask a remote computer to make a musically correct "inversion" of a polyphonic music stored as an audio file. On another hand, good audio time-stretching technologies are available, but all are relatively slow, and thus not very suitable for our real time purpose, especially in a web environment with files as big as the "pages" would be. Moreover, longer "pages" will result in yet bigger files, causing a much slower transfer rate. No one of these shortcomings is to be expected

[6] Standard mp3 compression does not appear to be a good idea for a very sonic-dependent music as is the "25 pages".

[7] By " reversion" we mean the "upside-down" score reading variable.

in a MIDI-based environment: pitches and durations can be very precisely manipulated with a few simple mathematical operations which are not supposed to consume a significant amount of time nor file size. Moreover, as stated above, there is no composer's directions for pitch, duration or intensity variables that could be "impossible" to apply to a MIDI file (see however 5th constraint).

- *Polyphony of pianos (up to 25) that may eventually play identical pitch(es) at the same time*. Here is another serious strong limitation of MIDI, because the polyphonic capacity of a MIDI device may fall down to 16 simultaneous pitches [8]. And as it is not realistic to assume the lambda user may have a set of two or more chained MIDI piano modules, there is also no possibility to have the same pitch played simultaneously. However, both situations may only eventually occur when a large number of pianos are activated and play some pitches *exactly* at the same MIDI onset position. In several cases, this situation can be avoided or at least minimized (see how below). Thus, it cannot be considered to be a definitive sentence against the MIDI implementation. Anyway, the audio solution, which have no one of such limitations, would be a much better choice, if, again, the weight of such a number of simultaneous audio data was not prohibitive for a web interactive environment.

Due to that considerations, the MIDI option, despite its limitations, appears to be more suitable option, because of the present state of audio and web technologies, and average internet transfer rates. A secondary factor which favored this choice is that "25 pages" was yet MIDIfied, during a former student research project [9]. This constitutes a valuable shortcut for the implementation to be quickly available.

Java implementation

For several reasons, SUN's Java appeared to be the better suited programming language for this application.

- It is a multi platform language.
- It has a wide support for MIDI files, offering various classes that allow manipulation, execution and/or creation of new MIDI sequences. Besides, these classes are well documented, allowing the programmer to use the available resources in an easy and efficient way.
- It allows the application to be accessed through the web, a *sine-qua-non* condition for our project.

Our implementation consists of a homepage and a Java application. In the homepage, the visitor declares:

1. The number of active pianos.
2. How many pages each piano (pianist) will play, and in what order.
3. For each page, the two performance variables:

[8] When an overflow of MIDI pitch data occurs (i.e. a flow of more than 16 simultaneaous events), the device uses some kind of priority algorithm, giving generally preference to the higher pitch, which is supposed to carry the melodic (i.e. main) meaning of the music, not a valid criterium for "25 pages".

[9] The MIDI files have been realized by Ernesto Trajano, a pianist now post-graduating in computers, and are already available on the GMT site http://www.liaa.ch.ufpb.br/~gmt.

3.1. The page position, with two options: "up" (the page is read in its common position — i.e. the MIDI file is normally played — or "down" (the page is read upside-down);
3.2. The treble/bass clef rule, also with two options: "normal" (upper stave in treble clef, lower stave in bass clef — it is the way the MIDI files are recorded) and "permuted" (upper stave in bass clef, lower stave in treble clef).
4. The duration of each page.
5. The position of the piano on the virtual stage.

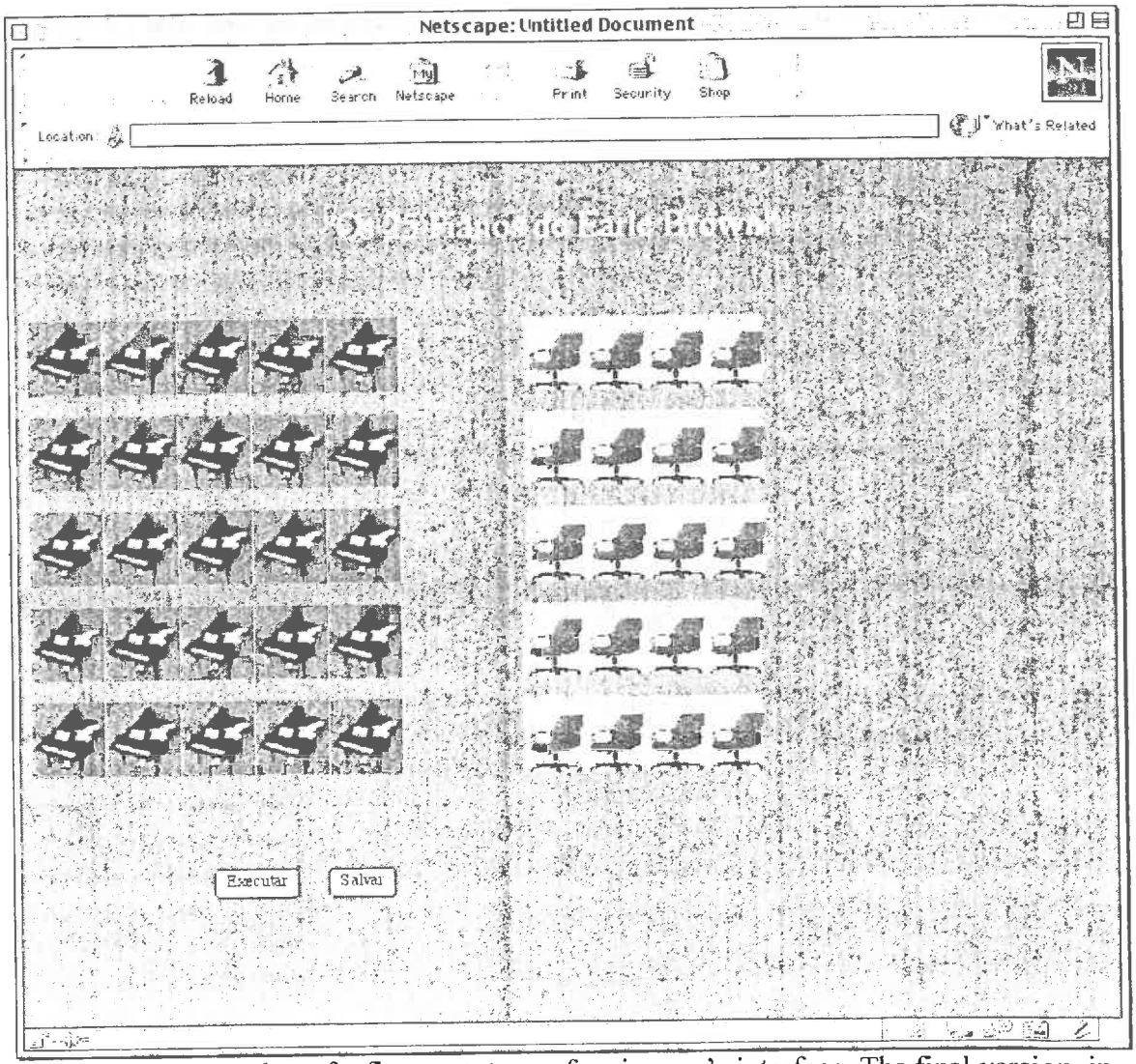


Fig. 2. : A screenshot of a first prototype of main user's interface. The final version, in english language, will represent a "true" theater. Each piano icon links to the programming page (see next fig.).

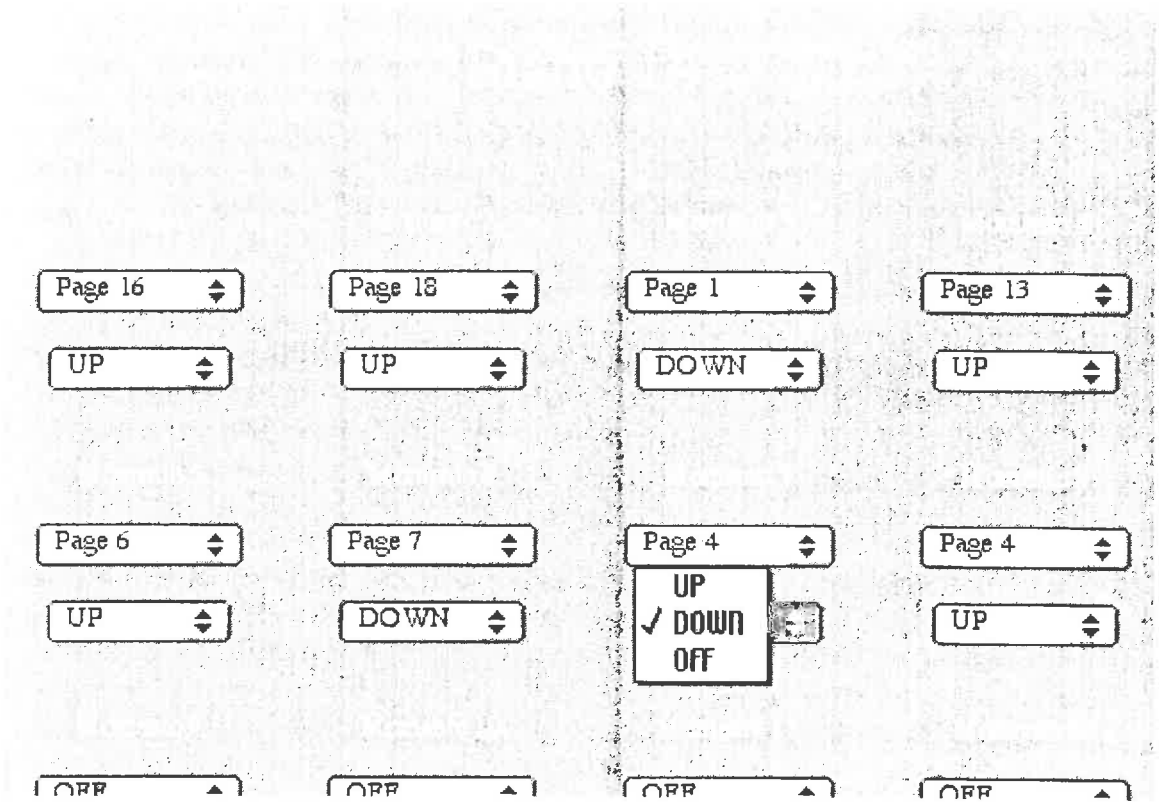


Fig. 3. A prototype of each piano interface (here the 14th piano), where the user informs what and how the piano will play. This interface is available for each of the 25 pianos the user wants to activate. At the present date (submission deadline), only the variables 1, 2 and 3.1 have been implemented.

Then the Java application executes the piece according to this program. The communication between the homepage and the application is made with JSP (Java Server Pages), allowing the parameters declared by the user to be passed to the application. The application has a main class, the *Manipulador*, that does all the manipulation of the MIDI sequences. It is responsible for the concatenation of the various sequences into one single sequence, for its execution, and for ending it. This class is also responsible for the reversion function (see below).

Discussion, problems and on-going developments

While the implementation of the 1st and 2nd performance rules is rather trivial, we find serious Java performance problems when executing the 3rd variable's "reversion" algorithm. At first, we planned to have the MIDI files to be reversed on-line, according to the user's declaration. As stated above, this task would be accomplished by a set of methods available in the *Manipulador*, using the onsets and pitches MIDI data input [10].

[10] The onset reversor takes the highest value (i.e. the last event in the file) as the starting point and play back. The pitch reversor works on the central value MIDI 60 and reverses each other values (MIDI 61 becomes MIDI 59, and so on).

But during the tests, we observed that the time consumed for these calculations is not adequate in a web environment [11]. This happens mainly because Java is an interpreted language [12]. Low or instable connection speeds can turn this problem even worse. Thus we resolve to build "by hand" a set of new 25 MIDI files corresponding to the "reversed" versions of the pages, so that the on-line task is limited to a pointing function to the corresponding MIDI file.

A second difficulty is due to MIDI specifications. The "clef permutation" variable is awkward to apply, at least if one want to strictly fulfill the composer's rule: « events within each 2 line system may be read as either treble or bass clef » (Brown *op. cit.*). The MIDI protocol does not have any kind of "clef data". We plan to implement this function in the following way: each page have a "pitch axis" (determined after a human analysis of the score); any pitch above this axis is said to be written/played in treble clef, while any pitch below, in bass clef. This constitutes the "normal" state of the already recorded MIDI files. If the user chooses the "permuted" version, an algorithm executes a double simultaneous transposition of pitches — the "treble clef" pitches are lowered 21 semitons, while the "bass clef" pitches are raised in the same proportion —, a relatively simple task. 21 semitons is the average interval a pitch which is written in a clef is transposed when read in the other clef [13]. We are aware we can face the same Java performance problem as for the reversion algorithm. Besides, this solution constitutes a limited interpretation of the original rule, as the composer admits both staves may also be played in the same clef. But the musical result would sound satisfactorily convincing.

The duration variable will be easily implemented by applying a multiplier to the MIDI onset and durations values. This multiplier will range from (0.5) to (2.00), up to the user's decision. Besides attending the composer's direction, this variable is also a powerful tool to avoid MIDI polyphony saturation, as a very fine tuning of the duration of each piano part can dramatically decrease the probability of several events to occur at the same onset position.

The last variable — position of each piano on a virtual stage — involves a rather complex MIDI manipulation of three parameters: the *pan* — the standard #10 MIDI controller —, the *main volume* — #7 controller — and a distance simulation through the MIDI control of room or reverb effects (like the #91 controller). The musical result that these controllers may produce depend exclusively on the user's MIDI device capabilities. Apart of the main volume, they are only fully implemented in professional devices. We are thinking about a custom-implemented controller. Anyway, this variable may be bypassed, or limited to pan and volume controls in a first version, as a spatial control of the performance is not explicitly requested by the composer.

---

[11] It took several minutes on the local computer.

[12] Compiled languages such as C++ e Delphi would have developped in this case a better performance.

[13] This is an average value, because, due to the diatonic structure of the piano keyboard, such "reading" transposition (not a truly "musical" one) is not evenly distributed. Upward transpositions from E and B, and downward transpositions from C and F, count only 20 semitons.

Finally, due to polyphonic MIDI limitations, the user should be advised to avoid to start the music with all pianos activated, and to connect more than one piano to play the same page at the same time. Nevertheless, as the exact behaviour of MIDI data flow depends on Internet conditions and on the user's local configuration, it may appear polyphonic saturations when a high number of pianos is activated. It means that some notes may drop or miss. As we have already suggested, a careful mapping of the duration variable will undoubtly help to avoid a great number of such shortcomings.

Anyway, the MIDI issues do not appear to invalidate the project as a whole, nor the musical experience it will certainly provide. Otherwise, we must thoroughly test the on-line Java performance, in order to study some kind of alternative if necessary. We will implement a "default" version of this piece, immediately executable by any new visitor. More, save/open functions should be implemented to allow, not only the user to keep recorded his own version on his hard disk, but also on the remote site, in order to other users be able to listen to and download.

This project is planned to be fully achieved and published till July 2001. It will be freely available at the GMT site: http://www.liaa.ch.ufpb.br/~gmt [14].

References and sources

[Brown 1974] Brown, Earle. "25 pages for 1 to 25 pianos". Toronto: Universal Edition, 1975 (originally composed in 1953, but the quoted text is dated "May 1974").
[Delaigue 1989]. Delaigue, Oliver. Les Nouvelles Musiques Americaines en France, 1945 – 85. Dissertation in Musicology, Paris: C.N.S.M.D.P., 1989.
[Leão *et al.* 2001] Leão, H.B.S., Guimarães, G.F., Ramalho, G.., e Cavalcante, S.V. Benchmarking Wave-to-MIDI Transcription Tools. Electronic Musicological Review, 2001, Vol. 6 N.1. http://www.cce.ufpr.br/~rem/.
[Nyman 1999]. Nyman, Michael. Experimental Music. London, Cambridge, 1999, (originally published1974).
[Ryan 1999]. Ryan, David. Earle Brown - A Sketch. http://www.earle-brown.org/, 11/1999.
Sun Microsystems. Application Programming Interface (API) Description, Java Sound Midi, Java Server Pages. Available at http://java.sun.com/products/jsp/.

[14] The Earle Brown's software is part of a main research project on 20th century music, coordinated by Didier Guigue and supported by the CNPQ (Brazilian Council of Research).

Virtualis, opéra interactif

Alain Bonardi
Université de Reims
alain.bonardi@wanadoo.fr

Francis Rousseaux
Université de Reims
francis.rousseaux@univ-reims.fr

Résumé

Dans cet article, nous présentons le projet d'opéra interactif sur CD-ROM *Virtualis*. Ce projet comporte une dimension scientifique aussi bien qu'artistique. Notre réflexion sur l'interaction entre un utilisateur et des contenus opératiques nous a conduits à utiliser des modèles de relations entre entités fondés sur des forces physiques, dont l'utilisateur est en quelque sorte absent. Nous détaillons quelques aspects de cet environnement de lecture mais aussi d'écriture sur des contenus artistiques complexes, entre texte, musique et graphiques.

Mots-clefs : opéra, interactivité, générativité, modèles de l'interaction.

1. Principes de *Virtualis*

1.1. Introduction

Virtualis est un opéra interactif sur support informatique (CD-ROM) que nous développons à la fois comme projet artistique de création et comme projet de recherche dans le domaine de l'interaction musicale homme-machine. Il offre à un spectateur la possibilité de jouer[1] non pas à l'opéra – il ne s'agit pas d'une reconstitution en réalité virtuelle d'une œuvre du passé ou d'une salle lyrique existante –, mais bien avec l'opéra, c'est-à-dire d'interagir avec des contenus lyriques. Dans le cas d'un opéra numérique tel que *Virtualis*, ces derniers recouvrent des textes (parlés ou chantés), des fragments musicaux et sonores, ainsi que des éléments graphiques.

1.2. Principes artistiques

L'opéra interactif s'inscrit dans la lignée des œuvres ouvertes proposées par les compositeurs dans les années cinquante et soixante. En 1957, Pierre Boulez (né en 1925) dans sa *Troisième Sonate* pour piano et Karlheinz Stockhausen (né en 1928) dans son *Klavierstück XI*, proposent au pianiste des possibilités d'enchaînements variables entre des sections clairement écrites : l'instrumentiste, en appliquant des règles combinatoires posées par le

[1] Pour l'instant, *Virtualis* n'offre pas de possibilités de « jeu d'opéra » en réseau.

compositeur, peut proposer des parcours différents dans l'œuvre d'une exécution à l'autre. Mais un autre compositeur, André Boucourechliev (1925-1998) va plus loin que la simple articulation de contenus pré-définis : dans *Archipel IV* (1971) pour piano, il propose au musicien de constituer dynamiquement la musique elle-même, en associant des schémas mélodiques à des schémas rythmiques en temps réel.

Ces œuvres mettent d'une certaine façon l'auditeur hors-jeu. En effet, elles offrent aux interprètes des possibilités de renouvellement de leur rapport à l'œuvre, tandis que le compositeur pressent une nouvelle façon d'écrire, qui insiste plus sur des processus à déployer que sur des contenus à jouer fidèlement. L'auditeur ne se sent pas concerné par ce style d'écriture, qu'il percevra difficilement, sachant qu'il faudrait écouter l'œuvre plusieurs fois pour en percevoir le renouvellement.

Dans l'opéra *Votre Faust* (1968), le compositeur Henri Pousseur et l'écrivain Michel Butor ont tenté de proposer des modalités d'interaction permettant au public d'infléchir le cours de l'histoire. Malheureusement, cette œuvre n'a pu être créée dans des conditions correctes et les moyens d'expression mis à disposition du public (vote, interventions à voix haute, etc.) ont suscité un vaste désordre, qui a définitivement éloigné l'œuvre des scènes lyriques.

Les formes ouvertes « traditionnelles » sont tombées en désuétude. Nous pensons que l'informatique multimédia peut permettre de redonner vie à ce genre, en le transformant au profit de l'auditeur et au détriment du musicien interprète : le premier va pouvoir manipuler des contenus musicaux que le second se contentera d'enregistrer, étant dessaisi de ces possibilités d'intervention. De plus, de notre point de vue, il est important d'affirmer que le CD-ROM peut être le support d'une œuvre musicale à part entière, et pas seulement du commentaire d'une musique comme le sont de nombreux titres multimédia.

2. Modèles de l'interactivité dans *Virtualis*

2.1. Principes généraux

Nous avons commencé notre formalisation par une modélisation de la représentation d'opéra en nous appuyant sur une méthode inspirée des sciences de l'organisation et notamment de MADEINCOOP [Zacklad 1993, Rousseaux 1995]. Ces travaux nous ont permis de dresser les listes des tâches, des agents (humains/machines), les modèles de résolution collective de problème, de communication et de coordination pour les deux situations de représentation -opéra traditionnel dans une salle lyrique, opéra interactif où le spectateur fait face à l'ordinateur [Bonardi 2000]. Nous présentons maintenant les principes d'interactivité que nous avons retenus :

- nous cherchons une interactivité qui ne se dirait pas comme telle. Les liens et icônes se déploient habituellement pour désigner l'interactivité et ses lieux. Nous souhaitons mettre au point des processus fluides où il est implicitement proposé à l'utilisateur d'intervenir, s'il le souhaite. S'il ne le fait pas, l'œuvre continue sa trajectoire, selon des données de base et

selon ce que l'ordinateur a mémorisé des parcours antérieurs. Il s'agit ainsi de ne pas interrompre le flux de l'œuvre, mais plutôt de l'orienter.

- comme dans toute application interactive, il se pose la question de la prise en compte de l'intention de l'utilisateur et de son couplage à la réponse de la machine. Dans *Virtualis*, nous avons essayé de mettre en œuvre un modèle non psychologique : les motivations et le comportement de l'utilisateur ne sont pas modélisés. Ce dernier est considéré comme un élément extérieur qui peut agir sur le système autonome que constitue l'œuvre interactive sans être explicitement pris en compte dans sa complexité psychologique. Pour ce faire, nous avons élaboré un modèle fondé sur des forces physiques.

L'opéra interactif *Virtualis* propose trois types de scène :

- des tableaux offrant des interactions ludiques sur des dialectiques de l'opéra. Ainsi, dans le tableau intitulé « Les mots et la mer », des rochers représentent des mots alors que la mer représente la musique, et selon le niveau d'eau, ajusté par l'utilisateur, les mots parlés sont plus ou moins altérés, leur contenu sonore étant progressivement transformé en contenu musical. Nous ne reviendrons pas plus avant sur ces tableaux dans cet article.

- des scènes de transition intitulées « parcours de la musique » qui donnent l'occasion au joueur d'évoluer dans un univers tri-dimensionnel où la musique est représentée sous forme de métaphores graphiques, permettant à l'utilisateur d'interagir avec des contenus musicaux associés à des objets tri-dimensionnels. Nous présenterons ces scènes au paragraphe 2.2.

- des scène de transition intitulées « le Récit », qui sont des courts moments narratifs interactifs entre les personnages. Ce sont les passages pour lesquels nous avons développé le modèle physique sur lequel nous reviendrons en détail au paragraphe 2.3.

2.2. Les « parcours de la musique »

Les « parcours de la musique » sont des transitions entre tableaux. Elles peuvent intervenir ou non, selon un choix aléatoire, entre ces derniers. Leur principe demeure toutefois identique à chaque fois, bien que les musiques qui y prennent part varient. Trois fonctions principales sont proposées à l'utilisateur :

- « Errer » à travers la musique, en se promenant dans un espace géométrique qui en représente certaines propriétés, dans un univers où désormais musique et paysage se confondent.

- Modifier la musique, représentée sous forme d'objets graphiques qu'il peut manipuler.

- S'orienter vers telle ou telle autre séquence musicale.

Examinons tout d'abord les principes de la « promenade » dans les contenus musicaux. L'utilisateur, en quelque sorte fixé à la caméra qui nous permet de voir le paysage, « vole » comme un oiseau dans une vaste cage cubique, dont l'idée provient de l'univers du surprenant

film *Cube*[2], réalisé par Vicenzo Natali (1999). Cet espace est occupé par des objets musicaux représentés par des volumes tri-dimensionnels plus ou moins cachés, aux formes variées. Ces objets sont des fragments musicaux monodiques ; ils peuvent par exemple provenir de l'éclatement d'un morceau polyphonique en lignes individuelles. La figure 1 montre une vue de dessus du dispositif.

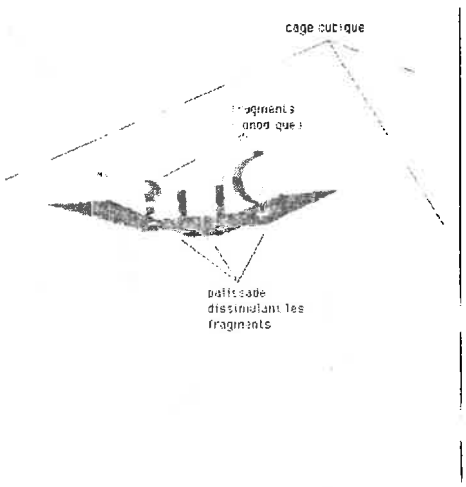


Figure n° 1. *Vue de dessus (en plongée) du dispositif des « parcours dans la musique ». L'utilisateur peut descendre et s'approcher d'un des quatre fragments monodiques, qu'il peut parcourir en détail.*

Les musiques associées à ces objets sont jouées en boucle, chacune ayant sa propre longueur, ce qui conduit à un décalage permanent entre elles. Les fichiers sonores se mélangent, mais les intensités relatives dépendent de la distance et de la position de l'utilisateur par rapport aux entités graphiques qui les représentent. Lorsqu'il se trouve à l'intérieur d'un volume associé à un objet musical, l'utilisateur peut décider de s'intéresser plus spécifiquement à lui. Par une commande, il choisit de se laisser porter par la musique, c'est-à-dire qu'il n'est plus libre de ses mouvements, mais est entraîné par le flot de la musique à travers l'objet qui la représente. La figure 2 ci-dessous montre un exemple de parcours dans un objet tri-dimensionnel ayant la forme d'un tunnel, associé à une mélodie jouée au piano.

[2] Dans ce film, un groupe de personnages se réveille dans un univers constitué de salles cubiques qui recèlent des pièges. Ils sont en quelque sorte prisonniers de la géométrie.

Figure n° 2. *Déplacement au rythme de la musique à l'intérieur d'une représentation d'un fragment monodique.*

Pour confectionner ces parcours, nous avons tout d'abord créé une application nommée ALMA [Bonardi & Rousseaux 2000], qui permet d'associer des objets graphiques à des contenus musicaux, de créer des variations mélodiques sur un objet donné et d'implémenter des interactions. Dans la version MIDI, l'utilisateur peut non seulement modifier des paramètres macroscopiques comme le tempo, le volume de restitution sonore, ou la position (panoramique) du son sur l'échelle stéréo de gauche à droite, mais aussi générer des variations structurales qui modifient les contenus joués. L'utilisateur choisit un fragment monodique associé à un objet graphique, et demande à l'ordinateur d'en générer des variations. Le système utilise le principe de recouvrement d'une entité mélodique par les motifs qui la composent [Baboni-Schilingi 1998].

2.3. Le « Récit »

Le « Récit » met en scène deux personnages, un homme et une femme, racontant une histoire donnée articulée en très courts « moments ». La question est de savoir comment rendre ce « Récit » interactif, et donc de choisir un modèle d'interaction pertinent. Nous avons choisi un modèle physique, utilisant des forces équivalent à la force d'attraction électrique et les champs de forces associés, en reliant, involontairement ou non, conjonctions amoureuses et conjonctions particulaires voire conjonctions planétaires[3], l'histoire ne disant pas si la pomme d'Isaac Newton avait le même goût que celle que croquèrent Adam et Eve.

Ce récitatif se déroule à l'écran dans un espace clos, où évoluent les deux personnages. Notre idée est celle d'un duo plus « chorégraphique » que « théâtral ». En effet, le point de vue de la danse, où le mouvement crée l'expression s'accorde bien au modèle physique retenu qui s'intéresse aux positions et vitesses. Nous préférons utiliser des poses sur des mouvements précis qui seront soit utilisées telles quelles, soit montées avec une certaine vitesse pour donner de courtes animations qui alterneront avec des phases statiques. Il s'agit d'intégrer ces aspects visuels au modèle de forces qui régit le « Récit ».

Les forces physiques sont exercées d'une part par chacun des deux personnages, d'autre part par des attracteurs situés en dehors de l'écran, produisant des champs de force supposés constants s'exerçant sur les personnages[4]. L'utilisateur/spectateur dispose des possibilités suivantes :

- Il peut d'une certaine façon diriger les chanteurs, en choisissant de donner un départ à l'un des deux chanteurs, en cliquant sur son icône. Mais il ne suffit pas de cliquer une fois, il faut régulièrement relancer le personnage dans son chant et son jeu, faute de quoi ce dernier s'estompe peu à peu visuellement et sur le plan sonore.

[3] Rappelons que les forces électrique entre deux particules chargées et la force de gravitation entre deux planètes ont des expressions analogues en $\frac{1}{r^2}$, r étant la distance entre les deux entités en présence.

[4] Sur le détail du modèle d'interaction, nous renvoyons le lecteur au paragraphe suivant.

- Il peut déplacer l'un des deux chanteurs, ce qui peut provoquer la modification de ce que celui-ci chante et le déplacement de l'autre chanteur.

- Il peut choisir un élément de décor menant à un autre court « moment » du « Récit » (sinon, par défaut, l'ordinateur se rendra au « moment » suivant) selon un principe de forces physiques qui sera explicité au paragraphe suivant.

- Il peut bifurquer vers un autre tableau.

Toute la dynamique du tableau est fondée sur le calcul en temps réel des forces et la mise à jour de la position et du chant des deux personnages. Nous supposons pour ce faire que quatre forces du type des forces électriques ou de gravitation s'exercent. Chaque personnage est donc modélisé par quatre « charges » ou « masses » que nous pourrions qualifier d' « affectives », placées selon quatre axes : aspiration à la tendresse, audace / résignation, égoïsme et jalousie. Elles expriment soit une grandeur positive, équivalent d'une masse, soit une grandeur (positive) et une nature (positive/négative), ce qui équivaut à une charge. Ces axes sont invariants, mais le poids de chacun des personnages sur chacun des axes change selon le moment du « Récit ».

Les interactions possibles concernent :

- l'attraction / répulsion entre ces deux personnages ;

- l'attraction / répulsion entre chacun des personnages et les forces extérieures.

La figure 3 ci-dessous montre les différents champs de forces et forces qui s'exercent sur la scène représentée par un rectangle. Les forces F_{21} et F_{12} sont respectivement celle qu'exerce l'homme sur la femme et son opposé. G_1 et G_2 quant à eux représentent des champs de forces s'exerçant sur les deux personnages selon les quatre « masses » ou « charges » que nous avons évoquées.

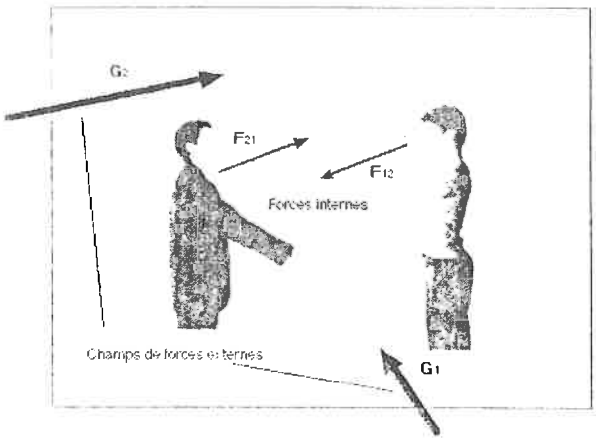


Figure n° 3. *Forces internes et externes s'exerçant sur les personnages.*

Pour ajuster les positions des deux personnages, l'ordinateur résout pour chacun d'eux l'équation fondamentale de la dynamique. L'algorithme d'Euler est utilisé deux fois pour une double intégration qui permet de calculer les coordonnées des deux modèles.

Le modèle de forces régit également la variation du texte et de la musique. D'une certaine façon, comme en danse, le mouvement crée l'expression. Chaque séquence chantée a donné lieu à plusieurs variantes à la fois du point de vue du texte et du point de vue de la musique. Les variantes textuelles ont été obtenues par glissement sémantique progressif sur un thème, en allant soit des sentiments intérieurs du personnage vers l'extérieur, soit dans l'autre sens. Sont ainsi constitués des axes sémantiques AS_i sur lesquels les forces extérieures au personnage auquel l'axe est associé ont plus ou moins de prise, selon un coefficient α_i compris entre 0 et 1[5].

Donnons l'exemple d'un moment dramatique qui se passe à la campagne. La thématique se résume en quelques phrases :

> L'homme et la femme passent l'après-midi à la campagne.
> L'homme aime ce lieu ; la femme s'ennuie. L'homme veut rester. La femme ne sait pas ce qu'elle veut. Elle soupçonne de ne pas être la première à venir avec l'homme en ce lieu.
> L'écran présente une gradation de gauche à droite : du plus campagnard à ce qui est le plus urbain, le tout rassemblé sur le même ruban qui peut défiler, selon l'endroit où se situent les personnages.

La figure 4 présente deux exemples d'axes sémantiques empruntés par les personnages.

[5] En fait, ces coefficients sont des cosinus d'angles compris entre 0 et 90 degrés, indiquant la direction de l'axe sémantique concerné, et permettant donc de calculer le produit scalaire des vecteurs unitaires de chaque axe avec chacune des forces.

Figure n° 4. *Deux axes sémantiques (à gauche, pour la femme ; à droite pour l'homme).*

Par le même procédé sont constitués des axes musicaux, AMj, de la mélodie initiale à sa variation la plus éloignée, également sensibles aux champs de forces extérieurs selon un coefficient β_j. Ces axes sont orthogonaux aux précédents, donc indépendants des niveaux sémantiques. L'ensemble articulé selon deux dimensions constitue donc un réseau musico-textuel, dont nous donnons un exemple ci-dessous, avec l'axe sémantique « elle me plaît aujourd'hui » croisé des variantes musicales. Pour chaque phrase (par exemple « tu as mis le haut que j'aime ») ont été composées (à « la main » ou avec l'aide de l'ordinateur, en utilisant l'environnement OpenMusic, logiciel du forum Ircam) plusieurs variations regroupées selon un axe vertical et classées de la moins à la plus passionnée.

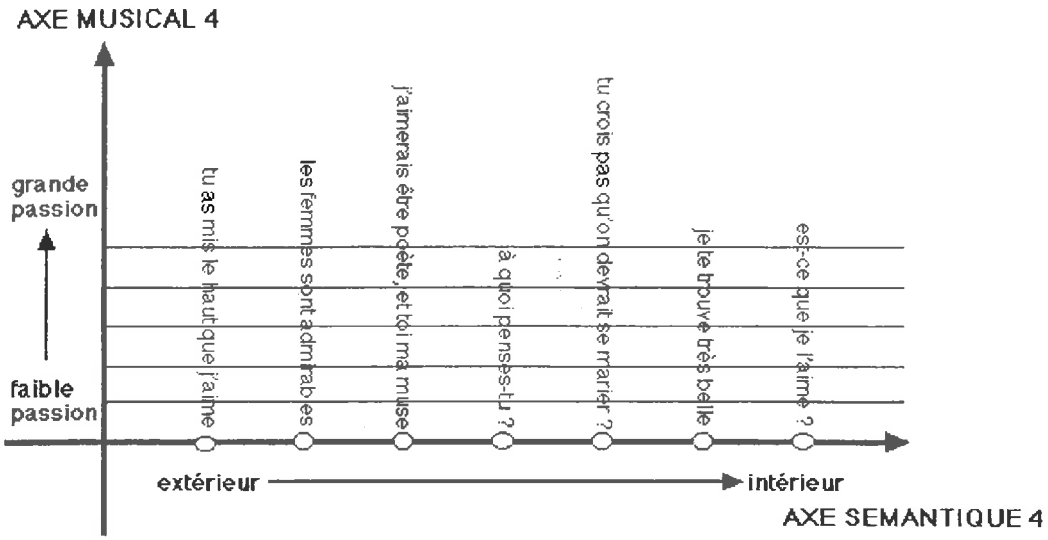


Figure n° 5. *Exemple de réseau musico-textuel.*

Voici par exemple (figure 6) trois réalisations musicales de la même phrase confiée au soprano (« moi, je vais tout lui raconter ») :

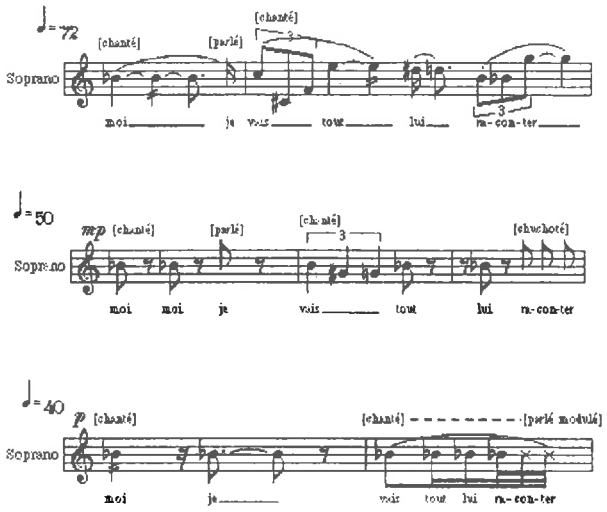


Figure n°6. *Trois exemples de réalisation d'une phrase confiée au soprano.*

L'orientation de ce réseau, et notamment le choix d'une progression plutôt selon le texte ou la musique est donné par une variable globale γ, sur laquelle l'utilisateur peut agir à tout moment, comme le montre la figure 7.

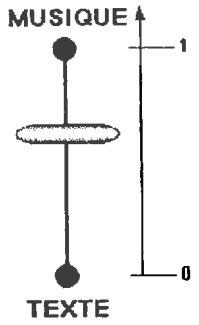


Figure n° 7. *Curseur de réglage des influences des champs de forces extérieures entre texte et musique.*

Cette variable varie entre 0 (tout texte) et 1 (tout musique). A tout moment l'axe musical AM_j où se trouve l'utilisateur dans le réseau, déjà doté du coefficient β_j est pondéré par γ et l'axe sémantique AS_i, déjà affecté du coefficient α_i , par $(1-\gamma)$.

Du point de vue de la réalisation, les différents fragments musicaux du réseau ont été enregistrées par une soprano et un ténor[6]. L'accompagnement des chanteurs a été généré par ordinateur grâce au logiciel de synthèse par modèle physique Modalys (logiciel du forum Ircam). Il est fondé sur des nappes sonores conçues pour fonctionner en fondu-enchaîné lors des changements de climat imposés par les changements de position des personnages et donc des changements de phrases chantées.

Ces modèles physiques mettent directement en relation les entrées gestuelles provoquées par l'utilisateur (mouvements de la souris, etc.) et l'actualisation des contenus. Contrairement aux modèles d'inspiration psychologique, ils abandonnent l'ambition de constituer un niveau supérieur d'interprétation des phénomènes physiques détectés. Ils ne tentent pas d'établir des faits déduits de ces actions, puis de décider de la réaction à apporter à ces faits « calculés », ce qui pose inévitablement le problème de l'interprétation conduisant à devoir expliciter le modèle de la machine pour qu'il adapte son comportement à elle.

Conclusion

Nous avons présenté le projet d'opéra interactif Virtualis et les modèles physiques d'interaction que nous avons été amenés à développer dans ce cadre. Pour l'utilisateur, ils constituent une activité originale de lecture, d'appropriation et d'écriture de « documents » lyriques, où l'absence de modèle psychologique suscite une relation singulière à une machine dont il ne s'agirait plus de deviner comment elle prend en compte l'utilisateur. Pour le créateur multimédia, les modèles de forces conduisent à une conception informatique qui n'est plus procédurale, mais, à l'instar de la programmation par contraintes, est fondée sur la définition de cadres interactifs au sein desquels l'ordinateur propose des solutions renouvelées d'une utilisation à l'autre.

---

[6] Nous voudrions remercier ici Sylvie Robert et Eric Gourouben.

Références

[Baboni-Schilingi 1998] BABONI-SCHILINGI, Jacopo, *Composition par modèles interactifs, systèmes génératifs et applications informatiques*, Mémoire de DEA, Formation doctorale Musique et Musicologie du XXe siècle, 1998.

[Barrière 1988] BARRIERE, Jean-Baptiste, « L'informatique musicale comme approche cognitive : simulation, timbre et processus formels », in *La musique et les sciences cognitives*, Liège, Mardaga, 1988, pp. 181-202.

[Bonardi 2000] BONARDI, Alain, *Contribution à l'établissement d'un genre : l'Opéra Virtuel Interactif*, thèse de l'Université Paris IV, 2000.

[Bonardi & Rousseaux 2000] BONARDI, Alain et ROUSSEAUX, Francis, *Alma, un environnement de « métaphorisation » de la musique*, Journées d'Informatique Musicale 2000 (JIM 2000), 15-18 mai 2000, Université Bordeaux I, Bordeaux (France), actes du colloque, pages 162-170.

[Bonardi & Rousseaux 1999] BONARDI, Alain et ROUSSEAUX, Francis, *Virtualiser l'opéra virtuel*, Journées ReViCo - Réalité Virtuelle et Cognition, Ecole Nationale Supérieure des Télécommunications de Paris (France), 14-15 décembre 1999, actes du colloque, pp. 21-31.

[Bonardi & Rousseaux 1998] BONARDI, Alain et ROUSSEAUX, Francis, *Premiers pas vers un opéra interactif*, Journées d'Informatique Musicale 1998, Agelonde (France), 4-7 mai 1998, Publication du Laboratoire de Mécanique et d'Acoustique du CNRS n°148, pages A4-1 à A4-7.

[Hoos & Hamel 1997] HOOS, Holger et HAMEL, Keith, *The GUIDO Music Notation Format Version 1.0 – Specification Part 1 : Basic GUIDO*, Darmstadt (Allemagne) : Technical Report 20/97 du Département Informatique de l'Université Technique de Darmstadt, accessible à l'URL : http://www.informatik.tu-darmstadt..de/AFS/CM/GUIDO/docu/spec1.htm.

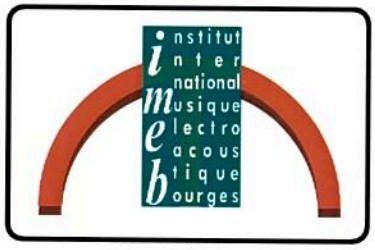
[Jorion 1990] JORION, Paul, *Principes des systèmes intelligents*, Paris, Masson, 1990, 188 p.

[Poizat 1986] POIZAT, Michel, *L'opéra ou le cri de l'ange*, Paris, Editions A.M. Métailié, 1986.

[Rousseaux 1990] ROUSSEAUX, Francis, *Une contribution de l'intelligence artificielle et de l'apprentissage symbolique automatique à l'élaboration d'un modèle d'enseignement de l'écoute musicale*, Thèse de doctorat de l'Université Paris 6, 1990.

[Rouseaux 1995] ROUSSEAUX, Francis, *Contribution à une méthodologie d'acquisition des connaissances pour l'ingénierie des Systèmes d'Information et de Communication : l'exemple de CHEOPS pour l'aide à la gestion de crises collectives à caractère géopolitique*, thèse d'habilitation à diriger des recherches, Université Paris VI, Paris, 1995.

[Zacklad 1993] ZACKLAD, Manuel, *Principes de modélisation qualitative pour l'aide à la décision dans les organisations : méthode d'utilisation du logiciel d'acquisition des connaissances C-KAT*, thèse de doctorat, Université de Compiègne, 1993.



INSTITUT INTERNATIONAL DE MUSIQUE ELECTROACOUSTIQUE DE BOURGES
1 Place André Malraux
BP 39 18001 BOURGES cedex
Tel : 02 48 20 41 87 - Fax : 05 56 84 66 69
Site web : http://www.gmeb.fr/

ÉCOLE NATIONALE SUPÉRIEURE D'INGÉNIEURS DE BOURGES
10 Boulevard Lahitolle
18020 BOURGES cedex
Tel : 02 48 48 40 00 - Fax : 02 48 48 40 40
E-mail : ensib@ensi-bourges.fr
Site web : http://www.ensi-bourges.fr