JOURNÉES D'INFORMATIQUE MUSICALE

JIM2012

GESTES, VIRTUOSITÉ & NOUVEAUX MEDIAS

9 - 11 Mai, 2012

UMONS, BELGIQUE

numediart

INSTITUTE FOR NEW MEDIA ART TECHNOLOGY

JOURNÉES D'INFORMATIQUE MUSICALE

JIM2012

GESTES, VIRTUOSITÉ & NOUVEAUX MEDIAS

9 - 11 Mai, 2012
UMONS, BELGIQUE

numediart
INSTITUTE FOR NEW MEDIA ART TECHNOLOGY

Préface du comité organisateur

Les Journées d'Informatique Musicale sont le rendez-vous annuel de l'informatique musicale francophone, un rassemblement de chercheurs et d'artistes venant de plusieurs pays sur plusieurs continents. Pour cette édition 2012 et pour la première fois dans l'histoire des JIM, c'est en Belgique, à Mons, que cet événement a lieu, plus précisément au sein de l'Institut NUMEDIART de l'Université de Mons.

L'implication de l'Université dans le domaine de l'informatique musicale ne date pas d'hier. D'abord, reconnue pour son expertise en acoustique des salles, la Faculté Polytechnique de l'UMONS a été le berceau de travaux pionniers en synthèse sonore dès les années 1990, puis plus récemment dans le domaine du contrôle gestuel de la synthèse. Aujourd'hui, les 50 chercheurs membres de l'Institut NUMEDIART pour les Technologies des Arts Numériques contribuent à développer cette discipline par des thématiques très diverses: les nouvelles interfaces, les instruments augmentés, les capteurs et outils pour la performance, la navigation hyper-média, la musique mobile, etc. Le paysage montois s'est également enrichi au fil des années de nombreux acteurs culturels et sociétaux: écoles d'art, ensembles de musique, salles de spectacle, et organismes de programmation et médiation, avec pour objectif commun la préparation de MONS2015 : Mons sera en effet capitale culturelle Européenne en 2015, sur la thématique "When technology meets culture". C'est cette importante dynamique locale qui permet aujourd'hui aux JIM 2012 de s'installer dans nos murs, et d'accueillir ses participants.

Pour cette édition, nous souhaitons mettre un accent particulier sur la thématique "Gestes, virtuosité et nouveaux médias" afin de revisiter le dialogue entre ces concepts très transversaux de l'informatique musicale, au regard d'une culture numérique en grande mutation. Ce débat conduit à la question récurrente de la virtuosité, ou plutôt des virtuosités, gestuelles ou conceptuelles, multiples face à la diversité des interfaces et des processus à maîtriser dans la performance numérique. Plus que jamais, l'explosion des nouveaux médias replace cette discussion entre gestes et virtuosité au coeur de l'actualité : qu'en est-il de la relation entre l'informatique musicale et la performance transmédia, de l'intervention de plus en plus importante des médias sociaux dans les formes d'expériences musicales et, par conséquent, de la redéfinition des frontières entre les différents acteurs de la performance musicale, et de ce qu'ils partagent?

Nous sommes très heureux d'avoir pu compter sur la mobilisation de la communauté "informatique musicale" pour la préparation et l'organisation de ces JIM2012. Nous remercions en premier lieu nos trois intervenants invités, Robert Normandeau, Marc Leman et Thierry De Mey, d'avoir accepté de se joindre à nous. Il nous faut également souligner la qualité et la disponibilité des auteurs, ainsi que le travail réalisé par le comité de lecture, dont la centaine d'évaluations a permis de sélectionner efficacement les articles. Les JIM 2012 résonneront enfin de quinze heures de concerts et installations, répartis sur trois jours. Nous souhaitons remercier les artistes et les organismes de programmation qui ont rendu possible cette composante très importante de la conférence. Pour terminer, un grand merci aux bénévoles de l'UMONS pour leur contribution à tous les niveaux!

Nous vous souhaitons une très agréable conférence.

Le comité d'organisation JIM 2012,
Thierry Dutoit, Todor Todoroff, Nicolas d'Alessandro

Conférence

Comité d'organisation

- Thierry Dutoit (UMONS)
- Todor Todoroff (UMONS)
- Nicolas d'Alessandro (UMONS)
- Christian Frisson (UMONS)
- Loïc Reboursière (UMONS)
- François Zajega (UMONS)
- Olivier Delgrange (UMONS)

Comité de pilotage AFIM

- Daniel Arfib
- Gérard Assayag
- Marc Chemillier
- Myriam Desainte-Catherine
- Dominique Fober
- Mikhaïl Malt
- Yann Orlarey
- François Pachet
- Laurent Pottier
- Julien Rabin
- Jean-Michel Raczinski
- Anne Sédès

Comité de lecture

- Olivier Baudouin
- Nicolas d'Alessandro
- Christophe d'Alessandro
- Myriam Desainte Catherine
- Olivier Delgrange
- Matthias Demoucron
- Stéphane Dupont
- Thierry Dutoit
- Jean-Julien Filatriau
- Dominique Fober
- Marc Leman
- Mikhail Malt
- Alexis Moinet
- Sylvain Pohu
- Laurent Pottier
- Loïc Reboursière
- Anne Sédès
- Damien Tardieu
- Todor Todoroff
- Caroline Traube
- Pierre Alexandre Tremblay

Concerts et installations

Coproduction

- ARTeM
- Transcultures (pour les Transnumériques)
- Le Manège.Mons
- Arts2 (l'Ecole Supérieure des Arts née de la fusion du Conservatoire Royal de Mons et de l'Ecole Supérieure des Arts Plastiques et Visuels)

Soutien

- Conseil de la Musique Contemporaine
- Commission des Arts Numériques de la Fédération Wallonie-Bruxelles

Aides

- Halolalune
- Musiques Nouvelles
- Charleroi/danse
- CECN
- Centre Henri Pousseur
- Espace Son Hainaut
- IPEM
- Logos Foundation
- La Médiathèque
- FAR Audio

PROGRAMME DES CONCERTS ET INSTALLATIONS

Concerts

9 mai à 19h30 - Salle des Arbalestriers (Rue des Arbalestriers 8)

Extension du corps sonore / Nouveaux Instruments

- Todor Todoroff, Laura Colmenares Guerra & Sigrid Vandenbogaerde: *eVanescens*
- Jean-Paul Dessy & Sigrid Vandenbogaerde: *Subsonic*
- Jean-Paul Dessy & Nicolas d'Alessandro: *Metastasis #2*

9 mai à 21h - Chapelle du Conservatoire (Rue de Nimy 7)

Electroacoustique sur orchestre de haut-parleurs

- Leo Küpper: *Digital Voices (Aviformes - Kamana - Paroles sur lèvres - Paroles sur langue)*
- Robert Normandeau: *Anadliad - Chorus - Eden*

10 mai à 19h - Salle des Arbalestriers

Conférence - Concert/Projection: La Trace du Mouvement

- Thierry De Mey: *Hands (extrait) - Silence must be! - Light Music - Tippeke (extrait) - Prélude à la mer (extrait) - Counterphrases (extrait)*

11 mai à 19h - Chapelle du Conservatoire

Piano et électronique

- Jean-Luc Fafchamps & Stéphane Ginsburgh: *Beth/Veth*

11 mai à 21h - Salle des Arbalestriers et Espace des Possibles

Live / Audio-Visuel

- Stephan Dunkelman & Angel Vergara: *Nous les oeuvres d'Art*
- Loïc Reboursière: *Quand deux vérités se rencontrent, que se disent-elles?*

11 mai > 22h - Soirée de clôture Espace des Possibles

- Pierre-Alexandre Tremblay & Sylvain Pohu (duo de type inconnu)
- Rafael Muñoz: *Set électro(acoustique) / DJ*

Installations

Du 9 au 11 mai: Maison Folie - Espace des Possibles

- Pieter Coussement: *Lament*
- Christian Frisson, Stéphane Dupont, Thierry Ravet, Xavier Siebert et al: *LoopJam*
- François Zajéga & Jean-Julien Filatriau: *HUM*
- Arnaud Eeckhout & Sébastien Herickx: *Ping Song*
- Sébastien Biset: *Bornes Archipel de La Médiathèque*

Le 9 mai: Maison Folie - Margin'Halle

- Logos Foundation: *Robot Orchestra*

PROGRAMME DE LA CONFÉRENCE

Conférenciers invités

Jour 1: mercredi 9 mai 2012

Session: Interfaces 1

Session: Analyse des Signaux Musicaux

Posters et Démos

Jour 2: jeudi 10 mai 2012

Session: Système et Languages

Session: Musique - Physiologie

Session: Outils pour la performance

Ateliers de programmations

Jour 3: vendredi 11 mai 2012

Session: Musicologie et Informatique

Session: Synthèse

Session: Interfaces 2

Session: Composition Assistée par Ordinateur

LA SPATIALISATION TIMBRALE OU LE MEDIUM, C'EST L'ESPACE

Robert Normandeau
Faculté de musique, B-406
Université de Montréal
robert.normandeau@umontreal.ca

RÉSUMÉ

Ce qui fait la spécificité du médium électroacoustique, c'est sa virtualité. Le son et la source ne sont pas soudés ensemble. Tel timbre entendu est complètement indépendant du haut-parleur à partir duquel il est entendu. Et celui-ci peut projeter toute la variété des timbres disponibles, ainsi que tous les autres haut-parleurs dans la salle. Et ainsi que tous les points situés entre tous les haut-parleurs.

Mais ce qui rend encore davantage unique l'expérience de l'espace en musique électroacoustique c'est la possibilité de fragmenter le spectre dans l'espace et en cela, elle se distingue nettement de la musique instrumentale. Là où nous avions un violon localisé à un endroit précis, nous avions aussi tout le timbre du violon. Alors qu'ici, il est possible de distribuer le timbre d'un son complexe en le répartissant sur l'ensemble des points virtuels disponibles. C'est ce que nous appelons la spatialisation timbrale: le spectre ne se retrouve en totalité que virtuellement dans l'espace du concert.

1. INTRODUCTION

Marshall McLuhan écrit en 1967 que le médium, c'est le message. Or la question qui se pose à nous aujourd'hui, soixante après son «invention», c'est quelle est la part du médium dans la musique électroacoustique, qu'est-ce qui lui appartient en propre et qui n'est pas redevable à la musique instrumentale? À notre avis, poser la question c'est y répondre : c'est dans l'apparition d'un art musical de support, à opposer à un art d'interprétation, que réside une part importante de la spécificité de l'électroacoustique. La communication d'aujourd'hui vise à présenter un aspect particulier de la musique électroacoustique, la spatialisation sonore et plus encore, la spatialisation timbrale, qui la détermine comme un médium unique.

2. L'ESPACE DU SON

Il est quelques exemples dans l'histoire de la musique où les compositeurs ont tenu compte de la notion d'espace dans la représentation de la musique. Mais dans aucun cas, la musique écrite n'a été réellement déterminée par la notion d'espace. Celui-ci est venu s'ajouter à fortiori, en prime en quelque sorte, comme un effet, spectaculaire parfois mais jamais essentiel. L'écoute stéréophonique, voire monophonique de ces musiques ne change en rien les valeurs musicales représentées, ni le sens du discours musical. Alors que dans la musique électroacoustique, cette notion sera développée au point que certaines œuvres, en dehors de leur contexte spatial, ne présentent pas beaucoup d'intérêt tant l'espace est déterminant dans leur composition même. Le médium, c'est l'espace.

2.1. Espace interne et espace externe

Traditionnellement, il y a deux types d'espaces en musique électroacoustique, l'espace interne, celui que le compositeur met dans l'œuvre, et l'espace externe, celui qui est ajouté en salle de concert. Le premier est fixé sur le support de l'œuvre et en fait partie au même titre que les autres paramètres sonores. Le second est variable et change selon les configurations de salles, de haut-parleurs, etc.

2.2. L'espace invariable

On peut cependant imaginer qu'il existe dans certaines œuvres un espace invariable où les espaces interne et externe seraient parfaitement confondus dans une relation «standardisée» comme la projection d'un film sur l'écran de cinéma et le son Dolby. Cette convention de représentation viserait alors à minimiser autant que faire se peut la part de l'acoustique locale dans la représentation de l'espace de la composition.

3. LA SPATIALISATION SONORE

Dès le premier concert de musique concrète la question de l'espace s'est posée avec toute son acuité. Comment représenter le son fixé en salle de concert? Depuis lors, de nombreuses stratégies de projection sonore ont été élaborées et expérimentées.

3.1. La projection stéréophonique sur des sources multiples

Les deux grandes approches de ce type de diffusion sont celles du Groupe de musique expérimentale de Bourges (GMEB) et du Groupe de recherches musicales de Paris (GRM) en France qui chacune à leur manière propose une redistribution de l'image stéréophonique sur un ensemble de haut-parleurs dispersés dans la salle de concert. Il s'agit d'un projection «manuelle» dans la plupart des cas, proche en cela d'un acte d'interprétation.

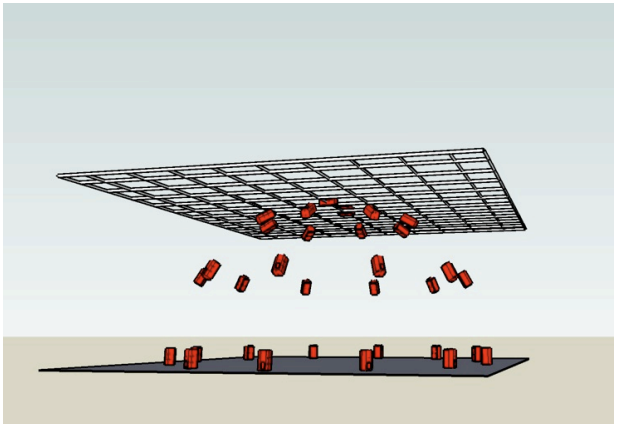
3.2. La projection multiphonique

Plus récemment, on a vu apparaître des musiques qui reflètent une spatialisation «composée» grâce à la technologie du multipiste. Ce type de projection se rapproche encore davantage du cinéma en ce qu'elle élimine complètement la notion d'interprétation pour reprendre à son compte l'idée du support fixe, y compris pour la question de l'espace.

4. LA SPATIALISATION TIMBRALE

On peut imaginer un compositeur de musique instrumentale plaçant ses interprètes un peu partout dans l'espace de la salle de concert. D'ailleurs on l'a vu abondamment depuis les années soixante. Mais ces œuvres seront toujours limitées aux timbres instrumentaux : le violon à gauche sonnera toujours comme un violon situé à gauche; de même la trompette à l'arrière.

4.1. La distribution du spectre sur des sources multiples



Ce qui fait la spécificité du médium électroacoustique, c'est sa virtualité. Le son et la source ne sont pas soudés ensemble. Tel timbre entendu est complètement indépendant du haut-parleur à partir duquel il est entendu. Et celui-ci peut projeter toute la variété des timbres disponibles, ainsi que tous les autres haut-parleurs dans la salle. Et ainsi que tous les points situés entre tous les haut-parleurs.

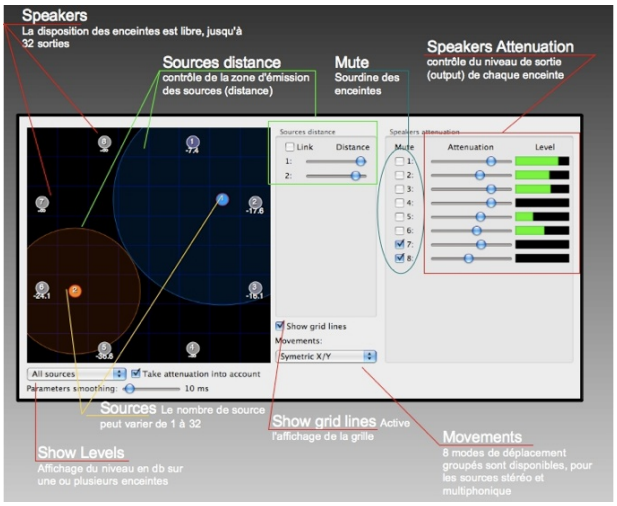
Mais ce qui rend encore davantage unique l'expérience de l'espace en musique électroacoustique c'est la possibilité de fragmenter le spectre dans l'espace et en cela, elle se distingue nettement de la musique instrumentale. Là où nous avions un violon localisé à un endroit précis, nous avions aussi tout le timbre du violon. Alors qu'ici, il est possible de distribuer le timbre d'un son complexe en le répartissant sur l'ensemble des points virtuels disponibles. C'est ce que nous appelons la spatialisation timbrale : le spectre ne se retrouve en totalité que virtuellement dans l'espace du concert. Chaque point de projection ne représentant qu'une fraction de l'ensemble. Cette projection, unique, exige que le compositeur ait eu à sa portée, au moment de la composition, un dispositif qui lui permette de fabriquer l'œuvre réellement. Ce n'est pas un espace ajouté à la fin de l'élaboration temporelle — comme c'est souvent le cas, surtout aujourd'hui où il suffit de prendre une session multipiste et de l'assigner à des sorties séparées — mais bien d'une spatialisation composée. Il s'agit bien d'un paramètre musical exclusif à l'électroacoustique.

5. LES OUTILS DEVELOPPES A L'UNIVERSITE DE MONTREAL

Le principe de base du développement de ces outils, au contraire de la plupart des outils autonomes qui existent sur le marché était de rendre possible la composition de la spatialisation en même temps que la composition du temps. Et non pas d'ajouter de l'espace après que la partie temporelle de l'œuvre ait été fixée. Il fallait des outils relativement simples et très économes en terme de CPU, puisqu'il fallait pouvoir les multiplier en très grand nombre dans des séquenceurs audio-numériques.

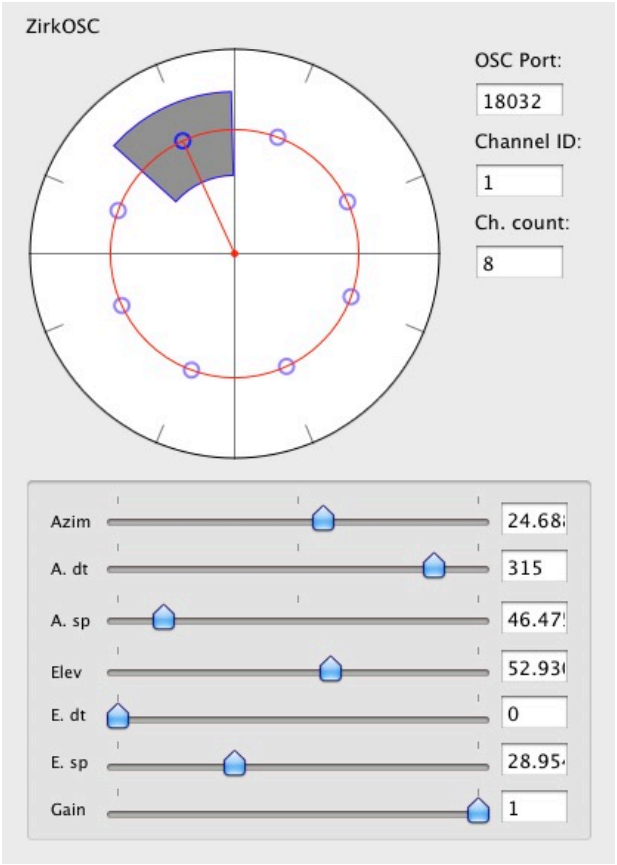
5.1. Octogris

Depuis quelques années, dans le cadre d'un programme de recherche structuré autour du Groupe de recherche en immersion spatiale (GRIS) nous avons développé deux outils qui permettent de gérer la spatialisation à partir de l'outil traditionnel de la musique acousmatique, à savoir le séquenceur audio-numérique. Nous avons constaté très rapidement lors de l'installation de notre studio octophonique au début des années 2000 qu'il n'y avait aucun séquenceur audio (sur Mac ou sur PC) qui permettait de synchroniser les pistes audio sur un simple orchestre de haut-parleurs octophonique!



Nous avons donc développé un plugiciel AudioUnit, l'Octogris, destiné à cela. Il permet de bouger des sources et le placement indépendant des haut-parleurs dans un espace bidimensionnel.

5.2. ZirkOSC



Dans le cas de pièces plus complexes qui font notamment appel à la spatialisation sur dôme de haut-parleurs comme ceux du ZKM ou de l'UdeM, il fallait développer un nouvel outil. Cet outil a été développé au ZKM en 2006 et il se nomme le Zirkonium. Mais celui-ci n'a presque pas de ressources de contrôle en temps réel. Nous avons donc développé récemment un second plugiciel AudioUnit, le ZirkOSC, qui permet, encore une fois à partir d'un séquenceur audio, de contrôler la spatialisation sur dôme. Cela ajoute la troisième dimension, la verticale, dans la gestion de la spatialisation sonore.

6. BIOGRAPHIE

Robert Normandeau est né à Québec (Canada), 11 mars 1955. Maîtrise (1988) et premier doctorat (1992) en composition électroacoustique de l'Université de Montréal. Membre fondateur de la Communauté Électroacoustique Canadienne. Co-fondateur de Réseaux (1991). Lauréat des concours internationaux de Bourges, Fribourg, Luigi-Russolo, Noroit-Léonce Petitot, Phonurgia-Nova, Stockholm, Ars Electronica (Golden Nica en 1996) et Giga-Hertz Preis (2010). Sa musique a été publiée sur de nombreux disques compacts dont sept disques solos, Lieux inouïs, Tangram, Figures, Clair de terre, Palimpsestes et le DVD Puzzles chez empreintes DIGITALes (Québec) et Sonars sur Rephlex (Angleterre). Prix Opus du Conseil québécois de la musique «Compositeur de l'année 98-99» et «Disque de l'année 1999» pour «Figures». Masques de la meilleure musique de théâtre décernés par l'Académie québécoise du théâtre pour Malina (2002) et La cloche de verre (2005). Il est professeur de composition à l'Université de Montréal depuis 1999.

Après avoir réalisé quelques œuvres instrumentales et mixtes, son travail de compositeur est aujourd'hui essentiellement consacré à la musique acousmatique. Plus spécifiquement, par les sonorités utilisées et les choix esthétiques qui la tendent, sa démarche s'inscrit dans un "cinéma pour l'oreille" où le sens tout autant que le son contribue à l'élaboration de ses œuvres. À son travail de compositeur de musique de concert s'ajoute parfois celui de compositeur de musique de scène, pour le théâtre notamment.

6.1. Œuvres principales

Matrechka (1986), La chambre blanche (1986), Rumeurs (Place de Ransbeck) (1987), Jeu (1988-89), Mémoires vives (1989), Bédé (1990), Éclats de voix (1991), Tropes (1991), Tangram (1992), Spleen (1993), Le renard et la rose (1995), Chat noir (1995), Figures de rhétorique (1997), Venture (1998), Ellipse (1999), Clair de terre (1999), L'envers du temps (avec Arturo Parra, 2000), Malina (2000), Jeu blanc (2001), Erinyes (2001), StrinGDberg (2001-03), Chorus (2002), Clair de terre II (2002), Clair de terre III (2003), Puzzle (2003), Éden (2003), Hamlet-machine with Actors (2003), Matériau pour Médée (2004), Palimpseste (2005), Palindrome (2005), Kuppel (2006), Murmures (2007), Pluies noires (2008), Jeu de langues (2009), Anadliad (2010), Like Radio (2010), La huppe (2011), La part des anges (2011).

6.2. Discographie sélective

Anthologie de la musique canadienne. Musique électroacoustique, Radio-Canada International (Canada). Cultures électroniques 3, Éditions Le chant du monde (France). Électro clips, empreintes DIGITALes (Québec). ICMC 96 (Hong Kong). Le petit prince, Radio adaptation de St-Exupéry, CBC Records (Canada). Prix international Noroit-Léonce Petitot 1991, INA•GRM/Noroit (France). Prix international Noroit-Léonce Petitot 1993, INA•GRM/Noroit (France). Prix Ars Electronica 1996, ORF (Autriche). Radius #4, What Next? Recordings (É-U. Storm of Drones, Sombient (É-U).

MUSICAL GESTURES AND EMBODIED COGNITION

Marc Leman
IPEM, Dept. of Musicology
Ghent University, Belgium
Marc.Leman@UGent.be

ABSTRACT

In this keynote, musical gestures will be discussed in relation to the basic concepts of the embodied music cognition paradigm. Video examples are given of studies and applications that are based on these concepts.[1]

1. INTRODUCTION

Interactive music systems have both a technological and experiential component. The technological component is concerned with the tools that translate human movements into manipulations of sounds. The experiential component is concerned with the intended control of the tools, which is accomplished through gestures. In this keynote, we mainly focus on the later component, as we believe that a good understanding of it is a prerequisite for a successful tool development.

Recently, the foundations of musical gestures have been reconsidered, and new viewpoints have been inspired by the insights from research on embodied music cognition (Godøy and Leman, 2011). The goal of the keynote is, firstly, to clarify the concept of musical gestures in relation to the basic concepts of embodied music cognition, secondly, to consider a number of implications for technology development (Leman, 2007). To illustrate that point, a number of video excerpts will be shown and discussed.

I should add here that in the paradigm of embodied music cognition, it is assumed that cognition is more than just mental processing and its corresponding brain activations. Indeed, music cognition is considered to be situated (i.e. embedded in an environment), and enacted (i.e. put into practice through action) (see Varela et al., 1991; Barsalou, 2008). The embodiment hypothesis refers to an action-oriented basis for making sense of our environment. In that context, we consider the former focus on music and mind within in a broader perspective that puts emphasis on embodiment, action, and consequently, gestures.

[1] This text is a modified version of a chapter that will appear in the book M. Lesaffre and M. Leman (eds.), "The power of music – researching musical experiences", Leuven: ACCO, 2012.

2. BASIC CONCEPTS OF MUSICAL EMBODIMENT

The paradigm of embodied music cognition is based on a number of concepts, related to: (i) the body as mediator, (ii) the gesture/action repertoire, (iii) the action-perception coupling, and (iv) the link with subjective experiences, such as intentions, expressions, empathy, and emotions. Let me introduce these concepts and discuss how musical gestures are related to them.

2.1. Body as mediator

The body of a person can be considered as the mediator between the person's environment and the person's subjective experience of that environment. This viewpoint is widely known (e.g. Merleau-Ponty, 1945), but it always needs a careful explanation. Indeed, the role of the body is particular in that it causally connects with a physical environment as well as with an experience (of the subject who owns that body). The physical environment can be described in an objective way (e.g. the waveform of the music on my sound recorder). The experience can only be described in a subjective way (my subjective feeling in response to the music on my sound recorder). Similarly, musical gestures can be described in an objective way as movement of body parts, but they have an important experiential component that is related to intentions, goals, and expressions.

2.2. The action/gesture repertoire

When the body acts as a mediator between our environment and the experience, it will build up a repertoire of gestures and gesture/action consequences. Such a repertoire is called a gesture/action-oriented ontology. It is the set of commands and their perceived outcomes that is somehow kept in memory and used for the next action and the interpretation of the next perceptions.

The repertoire is based on the idea that humans interact with the environment on the basis of actions, that is, movements that are made to achieve a particular goal, such as access to resources, or access to other persons. The repertoire can also be considered as the reservoir of experiences. This repertoire thus comprises connections between commands, sensations of the external world

through our senses (exterioception), but also sensations of our body movements (proprioception) and of our body state (interioception). Musical gestures form the core component of this repertoire.

2.2.1. Gestures

Interestingly, the concept of musical gesture applies both to sounds as well as to body movements (Godøy & Leman, 2010). For example, in a study on guqin performance (Henbing and Leman, 2007), it was possible to show that sonic gestures (identified in music) reflect sound-producing gestures (hand movements), and that these gestures can be understood as concatenations of more elementary gestural components. The rules for combining elements of this alphabet are defined by natural constraints (i.e. physical, physiological limitations) as well as cultural constraints (preferred movements). More recently, Desmet et al. (2012) studied clarinet playing gestures in relation to musical intentionality and expressiveness. Leman and Naveda (2010) studied Samba dance from the viewpoint of spatiotemporal representations.

2.2.2. Entrainment

The research on entrainment considers how resonant systems adapt their synchronization with each other. Traditionally, this work has a more exclusively focus on timing aspects in relation to synchronization, such as the tempo and phase during synchronization tasks in music playing or dancing.

2.2.3. Gestures and entrainment

In several more recent studies conducted at IPEM, we consider the idea that gestures may condition entrainment, and therefore, that the focus of entrainment should be broadened to include a spatiotemporal dimension that is rooted in bodily gestures. Such an approach may lead to a gestural topology of point clouds, a concept that has been introduced by Naveda and Leman (2010) and further developed in Maes et al. (submitted). Typically, with gross motor gestures that subsume a particular repetitive choreography, there is a large spatial region where the body part can be at a particular point in time, which ensures tolerance for synchronization variability, and therefore some flexibility in entrainment.

In contrast with these gross motor gestures, fine motor gestures require a more precise spatiotemporal deployment. This happens when people interact with a music instrument. In that case, there is less tolerance for synchronization variability, and less flexibility in entrainment. Just consider the fact that rehearsals of music ensembles often aim at getting the synchronization right.

2.3. The coupling of action and perception

From the previous paragraph it is clear that the gesture/action repertoire forms part of a more complex mechanism that controls the interaction between environment and subjective experience. This mechanism, also called, the action-perception coupling system, or the action-perception engine, is responsible for prediction and, as we will see, also for issues that involve musical intentions (musical action-goals).

Understanding the components and the dynamics of the action-perception coupling system is a hot research topic in modern cognitive science (see Wolpert et al., 2011). In general terms, it makes sense to consider at least two mechanisms for learning, which I call the sensorimotor loop and the action-perception loop. The sensorimotor loop is a low-level loop where the motor activity is basically driven by sensory input from the environment. In contrast, the action-perception loop is a high-level loop that involves the gesture/action repertoire. The two loops are not exclusive and they can run in parallel. For example, in producing a sound on the clarinet, the sensorimotor loop can maintain the control of the mouth and breath in relation to the sound production, while the action-perception loop can use the repertoire of learned fingering patterns (that have been practiced earlier) as a global controller for the expressive production of the sound. As such, the assumption is that action and perception are controlled by an action-perception coupling system involving different loops.

Interestingly, the prediction principle also works the other way around. When a person perceives a certain sound, the person may rely on previously learned action-outcome knowledge in order to assume the action for the sound. The latter principle is indeed fundamental in our approach because it explains why interaction is embodied. This principle, which is comparable to a Bayesian inference in statistics, also offers a key to modeling our multimodal (sound-gesture) interaction with music. Typically, when an every-day sound is heard, the person will tend to act in relation to how that sound is produced. In contrast, when an abstract sound is heard, the person will tend to act in relation to particular parameters of the sound that can be reproduced by movements (such as general contours).

2.4. Musical intentionality, expressiveness, and empathy

Our research at IPEM has contributed to the idea that music may establish an intentional layer of communication between listener and player. This intentional layer provides access to the (assumed) goals of the music. It is based on an action-perception coupling system that tends to use actions as causes of perceived patterns, thus turning perceived sound patterns into action patterns that could have caused the perceived sounds. This mechanism forms the basis of a number of social phe-

nomena of music, ranging from empathy to social bonding.

3. THE ROLE OF TOOLS

Finally a word should be said about tools. Tools are of outmost interest to us for two reasons. Firstly, with tools, we can study the characteristics of the action-perception coupling system. For example, during piano playing, the action-perception coupling can be disturbed by using a headphone that introduces an auditory delay. In order to be able to cope with the music playing, the agent may have to change the usual strategy for predicting the consequences of his actions. The change of strategy may reveal the mechanisms that underlay the coupling system. Secondly, with tools, we can build new applications that intervene with the action-perception coupling system. One example is DJogger (Moens et al., 2010) that extracts the tempo of your walking and provides music at the same tempo of the walking tempo. Humans tend to entrain and thus walk in phase with the musical pulse. Typically, they minimize the prediction error of the walking pulse until it fits with the musical pulse. IPEM has developed tools for different domains such as music education, therapy or entertainment. During this keynote, several examples will be given and further discussed

4. CONCLUSION

The paradigm of embodied music cognition provides an interesting viewpoint on musical gestures. It assumes that music is based on a tight relationship between sounds and experiences that are mediated by the body. Apart from the research that aims at understanding this phenomenon, the goal is to be able to control the nature of this complex phenomenon, and apply this control in applications for music education, health, art and other domains.

The paradigm of embodied music cognition provides an approach of how to deal with the experiential component in interactive music systems. The research field is likely to become an important partner of technology-driven music research.

5. REFERENCES

Barsalou, L. W. (2008). Grounded cognition. Annu. Rev. Psychol., 59, 617-645.

Desmet, F., Nijs, L., Demey, M., Lesaffre, M., Martens, J.P., Leman, M. (2012 - in press). A statistical analysis framework for assessing a clarinet player's performer gestures in relation to locally intended musical targets. Journal of New Music Research.

Godøy, R. I., & Leman, M. (Eds.) (2011). Musical gestures : sound, movement, and meaning. New York (N.Y.): Routledge.

Henbing, Li., & Leman, M. (2007). A gesture-based typology of sliding-tones in guqin music. Journal of New Music Research, 36(2), 61-82.

Leman, M. (2007). Embodied Music Cognition and Mediation Technology. Cambridge, MA: MIT Press.

Leman, M., and Naveda, L. A. (2010). Basic gestures as spatiotemporal reference frames for repetitive dance/music patterns in Samba and Charleston. Music Perception 28, 71‑91.

Maes, P.-J., Amelynck, D., Lesaffre, M., Arvind, D., Leman, M. (submitted). Conducting Master: An interactive, real-time gesture monitoring system based on spatiotemporal motion templates.

Merleau-Ponty, M. (1945). Phénoménologie de la perception. Paris: Gallimard.

Moens, B., van Noorden, L., Leman, M. (2010). D-Jogger: Syncing Music with Walking. In Proceedings of the SMC conference 2010, pp 451-456.

Naveda, L. A., and Leman, M. (2010). The spatiotemporal representation of dance and music gestures using topological gesture analysis (TGA). Music Perception 28, 93-111.

Varela, F. J., Thompson, E., & Rosch, E. (1991). The Embodied Mind: Cognitive Science and Human Experience. Cambridge, MA: MIT Press.

Wolpert, D. M., Diedrichsen, J., & Flanagan, J. R. (2011). Principles of sensorimotor learning. Nature Reviews Neuroscience, 12(12), 739-751.

BIOGRAPHY OF THE AUTHOR

Marc Leman is "Methusalem" research professor in systematic musicology and director of IPEM, Dept. of musicology, at Ghent University. He has a background in musicology, philosophy and computing. He has about 300 publications, including the books "Music and schema theory" (Springer, 1995), and "Embodied music cognition and mediation technology" (MIT Press, 2007). He did pioneering work in the epistemological and methodological foundations of computational modeling and embodied music cognition. In 2007 he became laureate of the Methusalem for his project on musical embodiment. During the free time (if any) he plays trumpet in jazzcombos.

LA TRACE DU MOUVEMENT

Thierry De Mey
Charleroi Danses, Belgique
thierry.demey@skynet.be

RÉSUMÉ

Pour la séance de lecture / concert / projection, les pièces suivantes seront montrées :

- *Hands* (extrait) (stereo)
- *Silence must be !* (5.1)
- *Light Music* (8.1)
- *Tippeke* (extrait) (8.1)
- *Prélude à la mer* (extrait) (5.1, triptyque)
- *Counterphrases* (extrait) (5.1, triptyque)

D'autres documents illustreront la présentation. Ce document propose des extraits d'entretiens qui étayent le propos de Thierry De Mey.

1. ENTRETIENS

1.1. Extraits du cahier spécial de MOUVEMENT consacré à Thierry De Mey

Y a-t-il un vocabulaire, un système que vous mettez en place ? Peut-on parler d'un répertoire de sonorités, de gestes et d'images, qui alimenterait votre travail et dans lequel vous piocheriez ?

Est-ce parce que tout mon trajet s'est tracé dans le voisinage de la danse, du mouvement ? Même si j'ai également écrit pour des ensembles et des orchestres, je pense beaucoup la chose musicale en termes de mouvements. Et je n'en ai pas fini avec cela, puisque je suis en train de penser plusieurs systèmes pour noter *Light Music* (2004) et de voir jusqu'où pousser cette corrélation entre mouvement et musique. J'ai créé des patterns pour des danseurs, des partitions dans l'espace, mais je n'ai jamais véritablement noté les mouvements des danseurs.

En tant que compositeur, j'ai toujours été intrigué par l'exploration du champ du geste musical. Quand un geste est-il générateur de mouvements ? Quand existe-t-il pour lui-même ? Dans *Musique de tables* (1987), des gestes produisent du son, mais il y a également des gestes qui existent pour eux-mêmes, souverainement chorégraphiques. Lorsque j'ai affaire à des percussionnistes très entraînés, ils n'envisagent souvent que la partie productrice de son. Or je fais très attention à ce qu'ils exécutent une figure dans sa pleine hauteur, le cercle par exemple. Ces questions sont fondamentales car elles traitent directement de l'écriture, du "sens" musical, de la "musique de la musique". Au croisement de systèmes symboliques et analogiques, par quelles opérations de réduction de la pensée musicale, par nature multi-dimensionnelle, arrive-t-on aux deux dimensions de la feuille de papier : la partition ? Partition qui elle-même renaîtra sous forme de musique, à la rencontre d'une vie, d'un mouvement : celui de l'interprète.

Wittgenstein dit que la beauté d'une phrase de Brahms, lorsque l'on ne peut plus l'expliquer par des mots, va induire un geste de la main qui continue le fait musical. Notre main prolonge le mouvement comme pour attraper un indicible, une poétique en soi. Tout cela touche de très près à la manière dont notre esprit fonctionne, dont il perçoit la musique et le mouvement… Toutes ces questions s'enchaînent, s'enchâssent.

Sauf à considérer qu'il s'agit d'une seule et même voie ! Cette fusion entre le visuel et l'auditif, on la trouve ainsi dans ce titre de Paul Claudel : L'OEil écoute.

J'ai quelque peu fait ma spécialité de ce rapport ! Un geste dansé qui marche bien sur une musique : cela relève d'une certaine évidence pour moi. Je vois tout de suite si cela fonctionne ou pas. Si cela ne marche pas, je peux mettre en branle les fonctions analytiques pour essayer de décrypter pourquoi cela ne marche pas. Parfois, j'aurai une réponse, mais la plupart du temps la formulation échoue. Je cherche toujours à circonvenir ce "truc" indicible. Le phénomène opère dans les deux sens. Si je vois une belle phrase de danse, j'entends tout de suite des rythmes, des sons. C'est un peu de la synesthésie qui se rapporterait au mouvement.

Dans ce dialogue permanent, que vous ont apporté les nouvelles technologies, celles-là mêmes dont on dit justement qu'elles n'ont pas de corps ?

Avec *Light Music* (2004), je me suis dit qu'elles pouvaient donner une nouvelle physicalité au son. Je dois avouer que les nouvelles technologies m'ont permis d'accroître mon champ d'investigation.

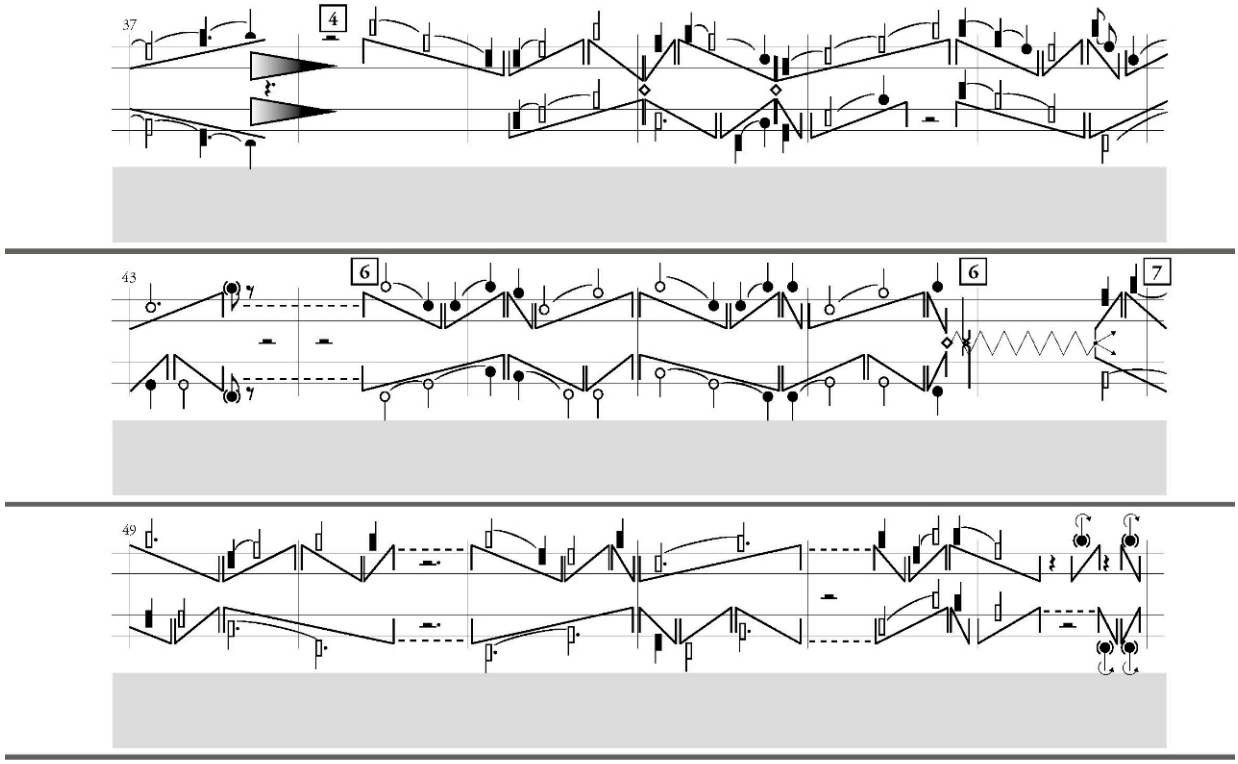


Figure 1: Extrait de la partition de *Light Music* (2004)

Après une représentation de *Light Music*, j'ai eu une discussion avec Pina Bausch, que cette pièce avait beaucoup touchée. Elle est venue dans les loges – j'étais très impressionné – pour me demander si j'avais fait ça avec le coeur, car, pour elle, c'était une pièce du coeur. Elle m'a alors avoué que, pour la première fois, les nouvelles technologies l'avaient bouleversée, parce que la générosité était présente... Je n'ai jamais pensé avoir inauguré une nouvelle forme d'art. De même, je n'ai jamais eu de frissons particuliers à l'idée d'employer des technologies militaires, policières. Ce sont des outils venus à point nommé dans mon champ. J'ai l'impression que la posture absolument low-tech, dans laquelle l'on se refuserait d'aborder les nouvelles technologies, est difficilement tenable. C'est un peu comme décider de vivre sans voiture, sans ordinateur, sans téléphone. On finirait par se retrouver comme un Robinson Crusoé au milieu de sa propre ville.

1.2. Extraits d'interview par Jean-Marc Adolphe et Charlotte Imbault: *L'informatique du mouvement*, analyse par Jérémie Szpirglas

«Ce qui m'intéresse, c'est la manière dont ces nouveaux outils ouvrent des brèches dans nos pratiques. »

Avec Thierry de Mey, tout est affaire de mouvement. Aussi, quand, en 2002, sa série de *Musiques de geste* se tait – au sens littéral du terme : l'exécution de *Silence must be!* se déroule dans un silence complet, il n'y a plus un son, seulement un chef d'orchestre qui, seul en scène, bat pour le public une polyrythmie proliférante, comme une machine bien huilée qui s'emballerait dans le vide –, l'aspect musical de sa recherche artistique semble être dans l'impasse. «A la sortie de ce travail, se souvient-il, j'ai eu le sentiment d'avoir touché du doigt une question essentielle pour moi, d'avoir atteint ce point limite où le geste musical – qui a pour objet de produire du son avec le moins de perdition d'énergie possible – cesse d'être fonctionnel et devient chorégraphique – donc susceptible de mettre en branle une musique de la musique, un mouvement musical dans la tête des auditeurs-spectateurs. J'étais arrivé au bout de ce parcours qui avait commencé avec *Hands*, puis *Musique de tables* et je ne savais pas comment passer à autre chose. Quand on a atteint la page blanche, le tableau blanc – cette fascination pour le trou noir que tous les arts portent en eux, à leur manière, comme une tentation – comment le dépasser ? » *Silence must be!* est à bien des égards un point de non-retour, qui s'accompagne d'un long moment de doute. Une fois la musicalité rendue au geste lui-même – jusqu'à le priver du son afférent – une fois le geste réhabilité dans sa musicalité propre, que faire ? Thierry de Mey envisagera même sérieusement l'abandon pur et simple de la composition, pour se consacrer entièrement à son travail de cinéaste.

C'est alors qu'il rencontre l'un des réalisateurs en informatique musicale de l'IRCAM, Laurent Pottier, alors au GMEM de Marseille. Laurent Pottier a vu *Silence must be!* et lui conseille d'aller voir du côté des technologies de captation du geste. Ces technologies, qui tâtonnent depuis les années 1980, commencent à l'époque à donner quelques résultats – notamment grâce à la croissance exponentielle de la puissance des ordinateurs.

« D'une certaine manière, cette rencontre avec ces nouvelles technologies d'informatique musicale a été salvatrice.»

En ouvrant de nouveaux horizons au rapport entre musique et mouvement, en faisant de l'interprète le générateur – en même temps que le manipulateur – du son électronique, en conservant à la musique cet aspect cinétique essentiel à ses yeux, la captation de geste a débloqué un processus créatif alors à l'arrêt. Et c'est ainsi qu'est né le projet *Light Music*.

Création commune de trois hommes – Thierry de Mey, bien sûr, mais aussi Christophe Le Breton, ingénieur au GRAME de Lyon, et le percussionniste Jean Geoffroy –, Light Music désigne autant la pièce elle-même que l'outil développé pour l'occasion : un instrument à part entière, qui allie capteurs de mouvement (accéléromètres et gyroscopes placés sur les poignets de l'interprète), reconnaissance vidéo du geste (notamment grâce à un « mur de lumière » qui agit comme un interrupteur on/off : si les mains de l'interprète sont en mouvement dans la lumière, l'ordinateur génère et/ou module les sons, sinon, il est inerte) et projection vidéo. Il y a ainsi un petit côté « J'en ai rêvé, l'informatique musicale l'a fait » dans les rapports qu'entretient Thierry de Mey avec la technologie. L'arrivée des premiers séquenceurs, qui lui évite de découper les bandes à la main, constitue pour lui une vraie joie. Puis, à l'heure des premiers balbutiements de l'écriture spectrale, le jeune compositeur s'y attelle avec enthousiasme – mais de manière parfaitement artisanale : il va même jusqu'à dessiner une gigantesque courbe logarithmique des fréquences sur les murs de son salon, tapissés de papier millimétré pour repérer partiels et autres harmoniques. Le calcul d'une simple progression harmonique lui prend la moitié du temps de composition de son premier Quatuor à cordes. Aussi, lorsqu'il arrive à l'IRCAM pour travailler sur *Amor constante más allá de la muerte* avec Anne Teresa de Keersmaeker, la puissance de calcul des ordinateurs est pour lui une bénédiction.

« C'était mon premier séjour dans ce grand magasin, se souvient-il avec une lumière enfantine dans la voix, et j'y découvrais tous ces jouets merveilleux. »

Il ne faut pas négliger en effet l'attitude foncièrement ludique avec laquelle Thierry de Mey aborde ces diverses technologies – qu'elles offrent une aide informatisée à la composition ou des traitements en temps réel de la matière sonore.

« Ce qui m'intéresse, c'est la manière dont ces nouveaux outils ouvrent des brèches dans nos pratiques. Face à un nouvel objet, mon premier réflexe est de jouer avec, pour tester ses limites, et voir jusqu'où il va m'emmener. »

Et, comme toujours chez de Mey, la clef de compréhension est invariablement le mouvement : un objet, qu'il soit artistique ou informatique, n'est exploitable que s'il résiste à la confrontation avec le geste – celui du danseur ou du musicien, ou même celui du spectateur (comme ce sera le cas dans l'installation *From Inside*, qui livrera au public l'instrument développé pour *Light Music*, en y ajoutant une petite épice venue des jeux vidéos).

« Avec le mouvement, tout est synthétique : mélodie, harmonie et rythme ne sont pas envisagés comme des entités distinctes. Un mouvement se passe dans l'espace et le temps, dans le mental et le corps. Et c'est par le biais du mouvement que j'aborde jusqu'aux aspects les plus strictement musicaux de mon travail : mouvement de zoom, de courbe, d'accélération… »

Kinok, musique de ballet pour ensemble instrumental (sans électronique), s'ouvre ainsi comme un travelling arrière : partant d'un gros plan sur un son multiphonique de hautbois, on élargit le cadre en répartissant le spectre de ce son aux clarinettes, puis en l'éclatant au reste de l'ensemble orchestral, vents et cordes.

« Le son est immobilisé, gelé (pour mieux l'analyser), puis projeté sur des durées musicales plus ou moins longues pour en faire des lignes qui peuvent évoluer dans le temps. Dans cette pièce, nous avons en outre imaginé d'associer chaque instrument à un danseur sur scène – et la torsion temporelle se reproduit ainsi dans la chorégraphie. Dans *Tippeke*, le principe sera plus ou moins le même, en partant cette fois de la voix d'Anne Teresa qui chante une comptine flamande. Je l'ai même élargi grâce au geste cinématographique : m'emparant de sons naturels, que je filtre très finement et précisément, j'en joue selon des techniques empruntées au lexique propre du cinéma. »

Cette démarche préhensible et ludique présente toutefois des dangers bien réels : on peut, d'une part, tomber dans le piège de la démo, où la présentation des possibilités d'un outil se substitue au projet artistique, et, d'autre part, on risque une uniformisation de l'esthétique.

« Il faut savoir dépasser la technologie, martèle Thierry de Mey, ne pas faire l'économie de ce moment où l'on rêve – où l'oeuvre est librement mentalisée, affranchie de toutes contingences technologiques ou techniques. On constate aujourd'hui combien les artistes ont tendance à faire l'impasse sur cette étape, au cinéma surtout : le numérique permet de filmer des heures et des heures – alors qu'auparavant, avec du film argentique, on pensait bien en amont ses plans et son story-board. »

« J'observe d'ailleurs, ajoute-t-il, que les concepts d'informatique musicale eux-mêmes mettent beaucoup plus de temps à se mettre en place que les technologies : nous travaillons aujourd'hui peu ou prou avec les mêmes processus qu'il y a quarante ans – nous les réalisons plus facilement, mais ce sont exactement les mêmes. Les nouveaux concepts sont excessivement rares. »

Chez Thierry de Mey, au contraire, l'outil informatique sait parfois s'effacer tout à fait. *Equi Voci*, par exemple, permet de réguler le rythme d'écoulement d'un film pour l'adapter au temps musical – sans contraindre le tempo et les musiciens (qui jouent en direct) au sein d'une structure temporelle trop rigide. Mêlant suivi de partition, calcul instantané du tempo et modulation vidéo, le but de ce procédé est justement d'être indétectable par le spectateur.

A ses yeux, l'informatique – et notamment l'informatique musicale – représentent ainsi une interface, avant toute autre chose. Interface entre le compositeur et le son, entre l'interprète et le son, entre le geste et la musique – comme dans *Light Music* –, entre le public et l'oeuvre. Et, pour cet artiste en constante recherche de pluridisciplinarité, l'informatique devient une réponse à son besoin d'interaction entre les disciplines : « Je sais d'expérience que si l'on additionne simplement les couches, sans les unir d'une quelconque manière, ces couches s'annihilent ou entrent en compétition. Je crois à l'art complet plus qu'à l'art total, à l'interpénétration des disciplines plus qu'à leur juxtaposition. »

2. BIOGRAPHIE

Thierry De Mey, né en 1956, est compositeur et réalisateur de films.

Une grande partie de la production musicale est destinée à la danse et au cinéma. Il fut souvent bien plus qu'un compositeur, mais aussi un précieux collaborateur dans l'invention de "stratégies formelles". Ses principales réalisations et compositions sont *Rosas danst Rosas*, *Amor constante*, *Kinok*, *April me* (chorégraphies Anne Teresa De Keersmaeker); *What the body does not remember* et *Les porteuses de mauvaises nouvelles* (chorégraphies de Wim Vandekeybus), *Musique de table*, *Frisking* pour percussions…

Les installations de Thierry De Mey où interagissent musique, danse, vidéo et processus interactifs ont été présentées dans des manifestations telles que les biennales de Venise, de Lyon et en de nombreux musées. Retenons entre autres *Deep in the wood*, *Light Music*, *Counter phrases*, *From Inside*, *Rémanences* et enfin *Equi Voci*, polyptique de films accompagné d'un orchestre philharmonique.

MEBIII, UNE INTERFACE DE SUIVI TRIDIMENSIONNEL DE POSITION EN CHAMP PROCHE A FAIBLE COUT

Roald Baudoux
Section de musique électroacoustique - Arts[2]
roald.baudoux@brutele.be

RÉSUMÉ

Notre activité musicale en studio et en live electronics nous a conduit à rechercher une interface gestuelle de suivi tridimensionnel de position adaptée à de très courtes distances. Après expérimentation sur de multiples interfaces commerciales et un travail avec divers capteurs, nous avons constaté le manque d'un outil à la fois fiable, abordable et adapté à nos besoins. Nous avons donc développé une solution personnelle sous le nom de *Mébiii*. Fondée sur la caméra infra-rouge d'une télécommande de console de jeu Wii, elle offre une résolution et une fréquence d'échantillonnage satisfaisantes. L'utilisation de ce capteur libère l'ordinateur de la tâche d'analyse des images et préserve donc la puissance de calcul pour le travail sur le son. Cette solution représente en outre un coût extrêmement réduit. Afin d'améliorer le suivi de la distance entre le capteur et la main, un circuit avec des diodes lumineuses est placé sur le dos de la main.

1. INTRODUCTION

Notre travail s'inscrit dans une double perspective : d'une part celle de la composition acousmatique en studio et d'autre part celle du travail sur scène en live electronics. Nous sommes essentiellement intéressé par des interfaces gestuelles de proximité.

Ce projet vise à la création d'une interface de commande gestuelle adaptée à un espace virtuel tridimensionnel de faible taille (50 x 50 x 50 cm approximativement). L'interface devrait être aussi réactive que possible, offrir une bonne résolution pour permettre de grandes nuances d'expression et ne pas nécessiter la prise en main d'un outil spécifique (tel qu'un stylet) afin de conserver la main libre pour basculer rapidement sur d'autres interfaces gestuelles (curseurs, souris, etc). En outre, nous souhaitons développer un système à faible coût d'une part parce que nous travaillons sur fonds propres mais surtout parce que nous voulons que ce travail puisse être partagé avec des étudiants aux ressources financières limitées.

Ce travail n'a pas obligatoirement pour vocation de constituer une recherche innovante mais plutôt de répondre pragmatiquement à un problème spécifique et avec un niveau de compétence limité dans le domaine de l'électronique.

2. QUELQUES INTERFACES EXISTANTES ET CRITIQUE

Nous ne prenons en compte que des dispositifs commerciaux disponibles à prix abordable ou des capteurs facilement intégrables dans des circuits électroniques.

2.1. Tablettes graphiques Wacom

Les tablettes graphiques sont aujourd'hui ubiquitaires dans le monde de la musique électroacoustique. Elles offrent au minimum trois degrés de liberté : position en x, position en y, pression. Certains modèles vont plus loin en détectant l'inclinaison du stylet. Cependant l'usage d'un stylet reste généralement nécessaire, ce qui implique en cours d'utilisation un temps pour prendre le stylet et un autre temps pour le déposer. Il existe bien des modèles capables de détecter la position d'un ou plusieurs doigts mais cette information n'est pour l'instant pas accessible dans le principal programme que nous utilisons pour la création musicale (Max/MSP) faute d'un développement logiciel adéquat. La contrainte du stylet peu gênante de prime abord le devient dans le contexte où de multiples interfaces sont combinées (curseurs, clavier MIDI, souris, etc). En effet dans cette situation la rapidité pour passer d'une interface à l'autre est un critère qui nous paraît important. En outre, la troisième dimension proposée sous forme de variation de pression travaille sur une échelle spatiale beaucoup plus étroite que les deux autres.

2.2. Magic Trackpad Apple

Le Magic Trackpad d'Apple offre une détection tactile tridimensionnelle de multiples points de contact. Le prix de l'appareil est très réduit (moins de 100 €) et sa calibration extrêmement bonne. De plus, il ne requiert pas de stylet. Outre la détection de position en x et en y, il autorise la détection de la surface de chaque doigt (jusqu'à… …onze !) en contact avec la surface. Cependant, l'appareil présente un double défaut : tout d'abord la résolution pour la détection de surface est très limitée. Enfin, la récupération de ces données par des applications musicales courantes n'est pas favorisée par le fabricant, qui communique un minimum

d'informations sur le sujet. Il existe un moyen de récupérer ces informations grâce à *fingerpinger* [1], un objet tiers pour Max/MSP. Il constitue une solution sophistiquée mais non fiable. En effet sans raison apparente il arrive que l'objet ne rapporte par les informations issues de l'appareil quoique celui-ci soit détecté par l'objet. Bien qu'extrêmement intéressante, cette interface doit donc être abandonnée.

2.3. Tablettes tactiles

Techniquement certaines tablettes permettent la détection de la surface de contact entre les doigts et la surface. Cependant cette information soit volontairement indisponible pour les développeurs (iOS), soit sa résolution reste limitée même lorsqu'elle est disponible (sur certaines tablettes fonctionnant sous Android). Quoi qu'il en soit la troisième dimension reste traitée comme un parent pauvre par rapport aux deux autres. Des technologies autorisant la détection de la pression en tant que telle sur des écrans tactiles ont été annoncées [2] mais elles ne sont pas disponibles actuellement.

2.4. Télémètres divers

Les télémètres par triangulation infrarouge représentent un système de détection de distance fiable et adapté aux distances recherchées. En plaçant de multiples capteurs sous forme d'une quadrillage on crée effectivement une détection sur trois dimensions avec cette fois la meilleure résolution pour la profondeur. Ce système est utilisé dans l'interface Peacock de Chikashi Miyama [3]. Cependant l'interface conserve la latence inhérente au principe de captation (triangulation optique) et qui est de l'ordre de 40 ms, ce qui est un peu trop pour un jeu rapide. D'autre part, la résolution en x et en y est limitée (on a 35 capteurs organisés en 5 lignes de 7 capteurs). Augmenter le nombre de capteurs ne résoudrait pas le problème car leur largeur est grande par rapport à la surface (±4 cm).

Les télémètres à ultrasons offrent un temps de réponse plus court (< 20 ms). Cependant, la détection sur trois dimensions exigerait l'utilisation de plusieurs télémètres, donc leur synchronisation et l'allongement des cycles de mesure par nécessité d'éviter les interférences entre capteurs, au minimum au nombre de trois. En outre l'utilisation sur une table pourrait souffrir des réflexions des ondes ultrasonores sur celle-ci et sur les autres objets posés à proximité (ordinateur, autres interfaces gestuelles, etc).

2.5. Détection capacitive

La détection capacitive offre en théorie la possibilité d'une détection tridimensionnelle et sans contact physique. Cette voie pose un certain nombre de difficultés de nature électrique et électronique que notre niveau de compétence en la matière ne permet pas de résoudre (blindage actif, conception des électrodes, linéarité, etc).

2.6. Caméras

La détection tridimensionnelle par caméra fonctionne très bien et ce depuis longtemps déjà [4]. Cependant l'analyse d'image demande généralement une puissance de calcul importante du côté logiciel (puissance d'autant plus grande que le nombre d'images par seconde est grand, ce qui peut s'avérer utile pour avoir une bonne réactivité) et c'est autant de puissance informatique perdue pour la création sonore, surtout lorsqu'on travaille avec des ressources limitées.

Certes l'arrivée de la Kinect de Microsoft a modifié le jeu et ouvert des perspectives. Néanmoins actuellement elle ne répond pas encore aux critères formulés. D'une part le taux de rafraîchissement est insuffisant (30 Hz) et d'autre part la distance minimale n'est pas adaptée (que ce soit dans la première version de l'appareil – 1,2 m – ou dans sa seconde – 0,5 m).

3. LA TELECOMMANDE DE LA CONSOLE WII

Cette télécommande a mis à la disposition du plus grand nombre un ensemble de capteurs facilement utilisables. La présence de certains d'entre eux est bien connue : accéléromètre, gyroscope. Cependant, un fait moins connu est que la télécommande contient une caméra adaptée à la détection des ondes infrarouges et destinée à aligner la télécommande dans le plan horizontal, pour lequel l'accéléromètre est « aveugle ». Les possibilités de ce capteur hors du champ du jeu vidéo ont été exposées par divers chercheurs, notamment Johnny Chung Lee [5].

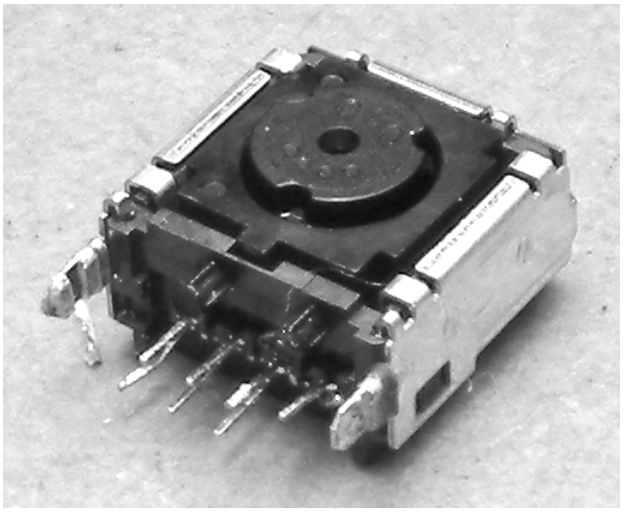


Figure 1. La caméra de la télécommande Wii

Cette caméra présente l'intérêt d'intégrer un microprocesseur d'analyse d'image capable de suivre la position de quatre points sur trois dimensions. Donc non seulement plusieurs points peuvent être suivis mais en outre et surtout le traitement d'image est réalisé dans le capteur lui-même et ne doit donc plus être pris en charge par l'ordinateur. En outre la fréquence de rafraîchissement des données s'élève à 100 Hz, ce qui est satisfaisant en termes de réactivité.

Les données des capteurs sont facilement récupérables sur un ordinateur par liaison sans fil Bluetooth. Cependant, pour placer la caméra à plat sur une table et l'orienter vers le haut (voir figure 2), il est nécessaire de l'extraire de la télécommande car elle est placée à l'une de ses extrémités latérales. Bien que son fabricant soit identifié (PixArt Imaging), elle n'est cependant pas disponible en tant que composant isolé et il et donc nécessaire de sacrifier une télécommande Wii pour l'obtenir.

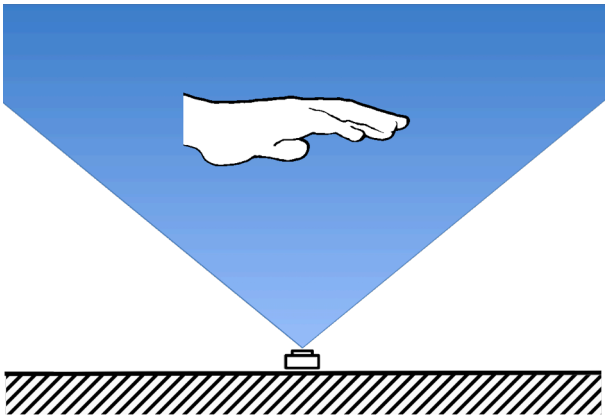


Figure 2. Configuration souhaitée pour la détection a priori.

Etant donné que la caméra utilise le protocole de communication normalisé I^2C, il est possible d'intégrer celle-ci dans un circuit électronique très simple et de connecter celui-ci à une carte avec microcontrôleur de type Arduino [6] et par là à un ordinateur. Tant l'identification des connexions de la caméra [7] que les codes nécessaires à la communication avec celle-ci [8] ont été déterminés par rétro-ingénierie et sont mis à disposition sur le Web par des tiers. Cependant, il s'avère qu'aux courtes distances prévues la caméra pose un problème d'angle de vue étroit (45° d'ouverture approximativement) qui limite fortement la taille de la surface de détection. Nous avons essayé de supprimer l'optique d'origine et de la remplacer par divers objectifs de substitution pour élargir l'angle d'ouverture mais sans résultat. Dès lors la seule solution consiste à éloigner la caméra de la main de plusieurs dizaines de cm, ce qui est incompatible avec un placement de la caméra sur une table. Il s'avère donc nécessaire de placer la caméra en position supérieure (figure 3). On en revient alors à une télécommande Wii standard reliée directement à l'ordinateur de destination par connexion sans fil Bluetooth sans qu'aucun circuit électronique ni carte Arduino ne soit requis.

4. LA MEBIII

4.1. Configuration générale

La télécommande est suspendue (par exemple à un pied de microphone) de manière à obtenir une distance d'environ un mètre au dessus de la table de travail. La caméra est orientée vers le bas et fait face au dos de la main. Afin de favoriser la détection, trois diodes lumineuses émettant dans l'infra-rouge sont placées sur un circuit, lui-même posé sur dos de la main au moyen d'une sorte de mitaine et alimenté par trois piles au format AA. Bien que le dos de la main soit en face de la caméra, des diodes lumineuses classiques avec un angle d'éclairement d'environ 30° se sont montrées trop sensibles à d'éventuelles inclinaisons même minimes de la main. Elles ont donc été remplacées par des diodes dites de haute puissance qui offrent un angle d'émission de 120°. Ces diodes (utilisées à puissance réduite) sont placées selon une configuration en triangle isocèle (voir figure 4) pour obtenir des distances différentes entre le point central et chacun des points latéraux d'une part et entre les deux points latéraux d'autre part.

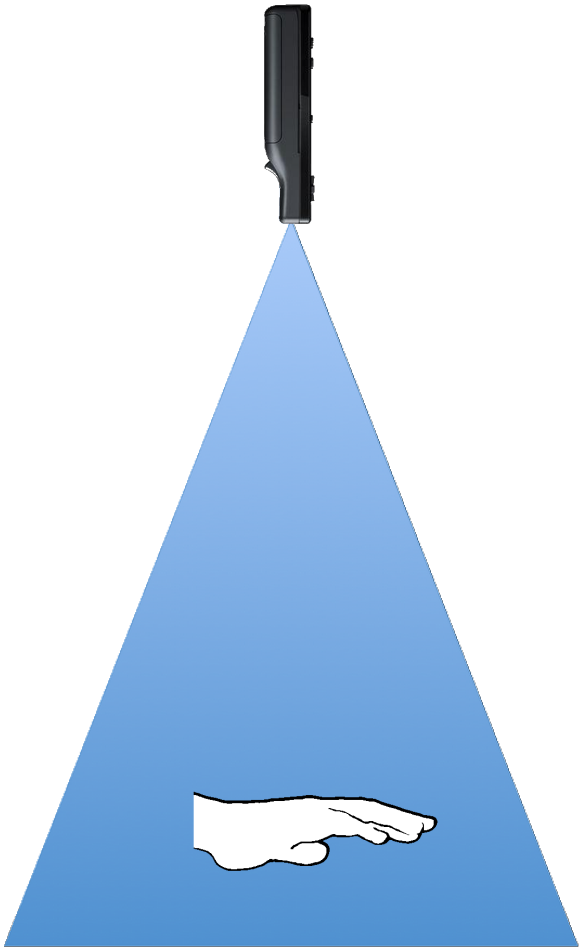


Figure 3. Configuration retenue après essais.

Cette différence de longueur permet de retrouver le positionnement relatif des trois points et de calculer l'angle de rotation de la main dans le plan horizontal, ce qui ajoute un quatrième degré de liberté à l'interface.

4.2. Configuration logicielle

La première version du dispositif traitait les données par une carte Arduino. La seconde reçoit les positions des trois points sur les axes x et y directement dans le logiciel Max/MSP via l'objet tiers *ajh.wiiremote* [9]. Nous avons développé un programme en langage Java directement dans Max/MSP via un module mxj pour effectuer les calculs nécessaires. La mesure de distance entre la main et la caméra fournie d'usine par la caméra elle-même possède une résolution très insuffisante pour une utilisation en finesse. C'est pourquoi nous avons décidé d'obtenir cette information d'une autre manière, à savoir via le calcul de la distance entre les deux points les plus éloignés l'un de l'autre dans le plan horizontal. La position globale en x et en y est déterminée par le calcul de la moyenne des positions des trois points.

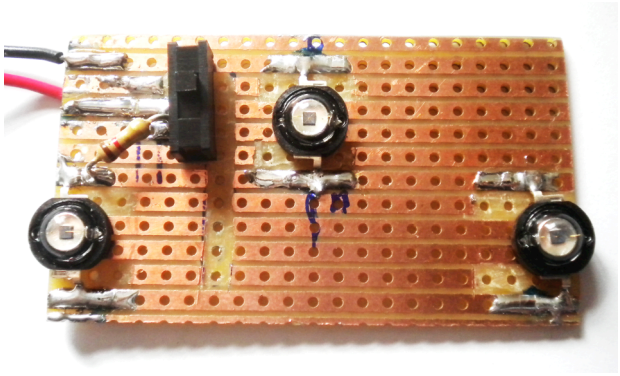


Figure 4. Circuit comportant les trois diodes lumineuses, à poser au dos de la main.

4.3. Application

La captation de position sur trois dimensions pourrait être appliquée au pilotage direct de trois variables directement sensibles telles qu'amplitude, fréquence de coupure de filtre et vitesse de lecture. Cependant notre approche est différente et vise à pouvoir coupler les avantages d'une modulation instantanée offerte par le geste à la richesse conférée par le temps de préparation sans limites du studio. Pour tout traitement appliqué sur le son (filtrage, distorsion, transposition, etc), nous associons donc à chaque sommet d'un cube virtuel une valeur mémorisée préalablement pour chacun des paramètres de ce traitement.

Le positionnement tridimensionnel capté via la Mébiii autorise une exploration de cet espace virtuel en créant par interpolation des réglages inédits et des variations continues sur de multiples aspects du son. Le geste est donc le moyen de réaliser de manière sensible un potentiel développé préalablement.

Le développement de la Mébiii est tout récent et d'autres utilisations, par exemple pour la synthèse ou le pilotage de divers modèles sont certainement à envisager.

5. CONCLUSION

Un capteur développé pour un appareil spécifique a été détourné de sa fonction d'origine en vue d'un suivi de position à très courte distance. La détection est couplée à un ensemble de diodes lumineuses émettant dans le registre infra-rouge pour améliorer l'estimation de distance par rapport aux données fournies directement par le capteur mais également pour autoriser le suivi de la position angulaire de la main. Le coût très réduit de la télécommande Wii et des développements additionnels rendent la Mébiii facilement accessible à des étudiants ou des musiciens qui ont une compétence technique limitée. L'interface offre un suivi de position sur trois dimensions couplé à une bonne fréquence de rafraîchissement. Cependant l'optique fournie d'usine rend l'utilisation à courte distance impossible. Un positionnement à une distance plus large que celle envisagée au départ et donc par dessus la main plutôt que par dessous doit donc être adopté.

6. REFERENCES

[1] Anyma, http://www.anyma.ch/2009/research/multitouch-external-for-maxmsp/ consulté le 28 février 2012.

[2] Anonyme, *« Notice Regarding the Signing of an Exclusive License Agreement with Peratech on the Use of a New Force Sensing Material »*, http://www.nissha.co.jp/english/news/2010/02/news-382.html, consulté le 28 février 2012.

[3] Miyama C., *« Peacock: A Non-haptic 3D Performance Interface »*, ICMC 2010 Proceedings, New-York, 2010.

[4] Merlier B., *« La main, le geste instrumental et la pensée créative - CG3D, Contrôleur Gestuel Tridimensionnel »*, Actes des Journées d'Informatique Musicale 2003, Montbéliard, 2003.

[5] Site personnel de Johnny Chung Lee, http://johnnylee.net/projects/wii/, consulté le 28 février 2012.

[6] RobotFreak, « Wii IR camera as standalone sensor », site Web http://letsmakerobots.com/node/7752 consulté le 28 février 2012.

[7] Kako, *« Component analysis of the infrared sensor for pointing the Wii Remote »*, traduction de la page Web http://www.kako.com/neta/2007-001/2007-001.html, consulté le 28 février 2012.

[8] Wiimote Wiki, http://wiki.wiimoteproject.com/IR_Sensor, consulté le 28 février 2012.

[9] Harker A. J., *AHarker Externals*, objects complémentaires pour Max/MSP, disponibles depuis le site http://www.alexanderjharker.co.uk/Software.html

VISION PAR ORDINATEUR ET APPRENTISSAGE STATISTIQUE : VERS UN INSTRUMENT DE MUSIQUE IMMATERIEL

[1]Sotiris Manitsaris *[2]Tsagaris Apostolos* *[2]Vassilios Matsoukas* *[2]Athanasios Manitsaris*
[1]Centre de Robotique (CAOR)
Ecole Nationale Supérieure des Mines de Paris (Mines ParisTech)
France
[2]Laboratoire des Technologies Multimédias et de l'Infographie
Département d'Informatique Appliquée
Université de Macédoine
Grèce
[1]sotiris.manitsaris@mines-paristech.fr [2]{tsagaris, vmats, manits}@uom.gr

RÉSUMÉ

Le présent article propose une méthodologie de vision par ordinateur pour la reconnaissance simultanée des gestes musicaux et complexes des doigts qui sont effectués dans l'espace sans instrument de musique matériel. Les techniques d'analyse d'images sont appliquées pour segmenter la main ainsi que pour détecter et identifier les doigts dans une vidéo. La reconnaissance et la prédiction des gestes des doigts se basent sur la modélisation stochastique des caractéristiques de haut niveau à l'aide des Modèles de Markov Cachés (MMC) et des Modèles de Mélanges Gaussiens (MMG). Des techniques d'invariance d'échelle et de rotation ont été appliquées. Cet article propose également une méthodologie de mise en correspondance des gestes des doigts avec des paramètres sonores pour le contrôle gestuel du son. Les applications concernent directement la composition de la musique contemporaine et en général les arts du spectacle.

Keywords : reconnaissance des gestes, modélisation stochastique, vision par ordinateur, contrôle gestuel du son

1. INTRODUCTION

Il y a quelques années le synthétiseur électronique présentait un concept révolutionnaire d'un nouvel instrument musical. Le synthétiseur produit des sons par génération de signaux électriques de différentes fréquences. Cet instrument musical a donné des nouvelles perspectives aux musiciens qui avaient déjà des connaissances au jeu pianistique. Pourtant, le clavier d'un piano, ou même celui d'un synthétiseur électronique, est constitué d'un mécanisme intermédiaire entre le musicien et la musique. Aujourd'hui, la nécessité de contrôler la musique en effectuant des gestes naturels dans un environnement réel devient de plus en plus importante. L'ordinateur peut désormais être considéré comme un instrument musical digital. Une fois que le geste est reconnu, plusieurs paramètres sonores issus de l'instrument peuvent être contrôlés de manière dynamique.

L'objectif de la recherche présentée ici consiste à proposer une méthodologie de vision par ordinateur pour la reconnaissance des gestes musicaux des doigts en temps réel sans instrument de musique matériel (*système PianOrasis:max*). La recherche actuelle est basée sur la méthodologie existante pour des gestes effectués sur un clavier de piano en temps différé (*système PianOrasis*). La méthode proposée permet la reconnaissance et la prédiction des gestes musicaux des doigts en utilisant des techniques d'apprentissage. La méthodologie est : (a) *Capable* de calculer tous les paramètres définissant les gestes des doigts ; (b) *Vision-orientée* vers l'image du musicien, sans analyse préliminaire ; (c) *Non intervenante*, permettant au musicien de se sentir libre, sans exigence d'équipement spécifique ; (d) *Accessible* et à faible coût, permettant l'utilisation à grande échelle. Deux différentes mises en œuvre de cette méthodologie sont faites. PianOrasis a été développé sur Matlab tandis que PianOrasis:max sur Max/MSP.

2. ETAT DE L'ART

Plusieurs recherches ont été menées pour la reconnaissance des gestes appliquée à l'interaction musicale en utilisant des techniques différentes. Ces techniques sont classées en trois catégories : a) l'approche basée sur la vision avec marqueurs, b) les interfaces tangibles et les capteurs embarqués et c) l'approche basée sur la vision sans marqueurs.

2.1. Approche basée sur la vision avec marqueurs

Les systèmes basés sur la vision à l'aide des marqueurs, tels que ViconPeak ou Optitrack, ont déjà été utilisé pour l'analyse de la marche, la rééducation de personnes handicapées ainsi que la réalisation d'effets spéciaux pour le cinéma d'animation en 3D [2]. Palmer (2000) a attaché des marqueurs réflectifs sur le vêtement

d'un pianiste afin de capter ses mouvements expressifs par mesure de déplacement des marqueurs [3]. Dans un autre cas, une recherche a été menée sur la mesure optique de capture des mouvements des violonistes en 3D, en utilisant le système Vicon 460 [4]. L'objectif de cette recherche a été la modélisation de l'interprétation musicale en obtenant des informations sur les mouvements du violoniste. Ces systèmes sont souvent utilisés pour la mise en œuvre d'une analyse du geste dans un temps différé par rapport à celui de l'interprétation musicale.

2.2. Interfaces tangibles et capteurs embarqués

L'information gestuelle délivrée en temps réel par les interfaces tangibles ou les capteurs embarqués [5], comme dans le cas de la manette Wii, est de très haut niveau. Même si cette technologie est souvent utilisée pour la reconnaissance des gestes effectués dans l'espace, il serait pourtant pratiquement impossible qu'elles soient appliquées dans la reconnaissance des gestes des doigts sur une surface ou un objet, puisque les musiciens se sentiraient extrêmement contraints.

L'IRCAM a développé un Réseau de Capteurs Sans Fil (RCSF) pour le suivi continu et la reconnaissance des gestes dansés et musicaux en temps réel [6]. Dans l'application du «violon augmenté», l'architecture tangible comprenait des capteurs d'accélération, des gyroscopes et un capteur de pression monté sur l'archet du violon, tandis qu'un bracelet autour du poignet du violoniste intégrait l'alimentation et l'émetteur sans fil ZigBee. Selon cette méthode, deux types d'informations sont mises en évidence de façon continue : (a) la similarité (*vraisemblance*) du geste effectué avec d'autres gestes préenregistrés et (b) la progression temporelle du geste effectué.

Des objets simples, tels que le ballon ou les échecs, ont été utilisés pour la création des interfaces sonores en utilisant un module central sans fil, qui comprend accéléromètre et gyroscope. C'est une approche à bas coût de type « do it yourself » plutôt, qu'une « prête à l'emploi » et l'utilisateur peut effectuer ses propres gestes pendant une partie d'échecs en y associant des sons de son choix [7].

2.3. Approche basée sur la vision sans marqueurs

Dans le cadre du projet iMuse (integrated Multimodal Score-following Environment), des techniques de reconnaissance des gestes par la vision sans marqueurs ont été utilisées. Le système reconnait les gestes d'un pianiste en temps réel et les synchronise avec la partition de la pièce. Une camera a été installée sur le clavier du piano, les gestes sont analysés durant l'interprétation et un graphique représentant les mouvements des doigts forme l'alignement avec la partition. Pendant les interprétations vivantes, les gestes sont suivis et sont comparés aux graphiques des gestes déjà enregistrés. La main du musicien est analysée d'une manière globale. Ceci dit, le système reconnaît plutôt les postures globales de la main que les gestes des doigts du pianiste.

Une autre approche est celle de type bio-musical, où il y a une source d'information, de type meta-musique, qui est placée sur le processus interne et externe et résulte en la création d'un contenu musical dynamique. Ce concept est transféré à l'expression musicale du cerveau et du corps, en utilisant le système EPOC EMOTIV (*électroencéphalogrammes*) pour l'acquisition des signaux du cerveau qui pourrait être utilisés pour la détection des émotions de l'artiste ainsi que la caméra Microsoft KINECT pour la reconnaissance des postures du corps de l'artiste [8].

3. METHODOLOGIE

La méthodologie se base sur celle proposée par Manitsaris (2010) sur la reconnaissance des gestes musicaux sur un clavier de piano [1] et vise à reconnaître les gestes musicaux des doigts qui sont effectués dans l'espace. Afin d'obtenir cet objectif, les bouts des doigts doivent être identifiés et localisés à l'aide d'une caméra bas coût. L'obtention des échantillons des pixels de l'espace RGB contenant de l'information de la peau et des ongles permet la création d'un modèle de peau (MP) la détermination de la Région d'Intérêt (RI) (Figure 1). L'application du MP permet la détection des régions de la peau sur n'importe quelle séquence d'images optiques.

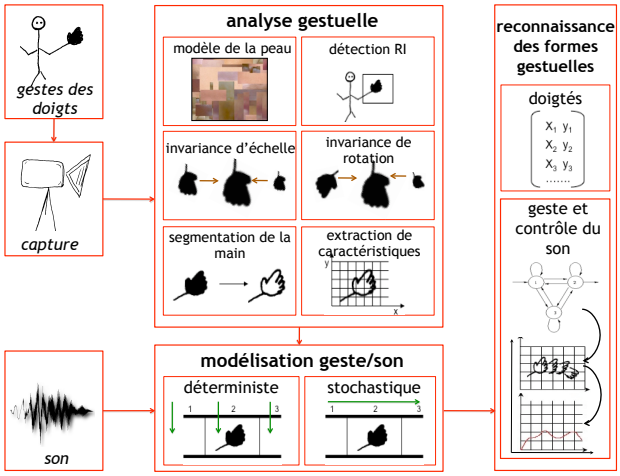


Figure 1. Méthodologie

L'étape suivante consiste à segmenter la main du musicien à l'aide des techniques de traitement d'images. Un ensemble de filtres sont appliqués à la séquence d'images afin de distinguer la zone de la silhouette de la main (*premier plan*) et le bruit (*arrière plan*).

Cependant, la problématique la plus intéressante reste l'identification des doigts. Plusieurs algorithmes pour l'identification individuelle des doigts sur l'image ont déjà été proposés. L'algorithme propose la détection des bouts des doigts en tenant compte des particularités de la posture de la main lorsque la paume est semi-étendue. Plus spécifiquement, la détection des bouts des doigts est effectuée en calculant les distances euclidiennes

entre le centroïde de la silhouette de la main et les coordonnées des pixels de son contour. Pour augmenter la performance de l'identification des doigts, des techniques d'invariance d'échelle et de rotation sont appliquées à chaque image de la séquence.

L'étape finale consiste à reconnaître les gestes. Un alphabet et un dictionnaire des gestes des doigts ont été créés. Les gestes étant des mouvements musicaux (*gamme ascendante/descendante, arpège, etc*) ont été analysés harmoniquement et mélodiquement afin d'extraire leurs états structuraux. Quatorze caractéristiques décrivant les gestes des cinq doigts sont extraits de chaque image. Ces caractéristiques des valeurs continues, autrement dites des vecteurs d'observation, ont été modélisées à l'aide des MMG, tandis qu'un modèle Markov différent est attribué à chaque geste musical [9, 10].

4. VISION PAR ORDINATEUR

La caméra est positionnée devant la main du musicien afin que les gestes des doigts puissent être enregistrés sans « doigt caché » en position de repos. Une vidéo de résolution maximale de 320x240 pixels est enregistrée.

4.1. Analyse et segmentation de l'image

Afin de rendre le système capable de détecter la peau dans une vidéo, un modèle de la peau (MP) a été développé. Par obtention d'échantillons des pixels de couleur de la peau et d'ongles extraits de la photothèque du pianiste (PP), la région d'intérêt (RI) a été déterminée. La normalisation de la RI, autrement dit la conversion de l'espace RGB vers l'espace normalisé rg, rend PianOrasis moins dépendant des variations de luminosité et permet d'identifier le MP en tant qu'un ensemble de valeurs. Le résultat exporté est une image binaire contenant soit la valeur 1 pour les pixels de peau (*avant-plan*), soit la valeur 0 pour le reste des couleurs (*arrière-plan*).

Par la suite, une séquence d'images binaires a été créée à partir de la vidéo importée, déterminant ainsi les régions contenant de l'information de peau et d'ongles dans l'image. Parfois, le MP n'étant pas parfait, de petites zones de l'arrière-plan sont considérées par le système comme si elles appartenaient à l'avant-plan et vice versa. Ce problème peut être résolu en appliquant des méthodes de traitement d'images (*morphologie mathématique et filtrage*) pour la réduction du bruit. Plus précisément, ces méthodes comprennent (a) la simplification de l'image binaire par réduction de bruit, extraction de la silhouette de la main et découpage de l'image binaire aux limites de la main et (b) la décomposition de l'image par extraction du contour de la main et des bouts des doigts [11].

Plus précisément, juste après le découpage de la RI aux limites de la silhouette de la main, des filtres Open, Min et Gauss s'appliquent de manière répétitive. Les paramètres des filtres déterminent le taux de leur implication à l'image. Suite à une recherche antérieure [12], le choix de ces paramètres se fait automatiquement en fonction du quotient du nombre des pixels de la peau sur le nombre des pixels de l'image.

On en déduit ainsi que le système est relativement indépendant des variations de la distance entre la main et la caméra (*invariance d'échelle*). En outre, le découpage de la RI aux limites de la silhouette de la main permet une analyse de l'image bien plus rapide [13].

4.2. Exportation des caractéristiques

La main du musicien prend une posture semi-étendue durant son interprétation, augmentant ainsi le niveau de difficulté dans la reconnaissance. Vue de face (*vue de devant pour la caméra*), la zone intérieure de la main étant également une région de peau, dans plusieurs cas la silhouette de la main est extraite en masse avec du bruit. En conséquence, la distinction des doigts dans l'image devient extrêmement difficile, surtout si la distance entre les bouts des doigts est très faible.

La localisation des doigts s'effectue en calculant les distances euclidiennes entre le centroïde et les coordonnées des pixels appartenant au contour des doigts. Dans le cas d'un « doigt caché », PianOrasis prévoit la position du doigt dans l'image suivante à l'aide des classificateurs, en tenant compte de la « mémoire du geste », calculée en continu par les positions des doigts dans les trois images précédentes.

À partir du moment où le centroïde est calculé et les bouts des doigts sont identifiés et localisés dans l'image, le système peut extraire les vecteurs d'observations, en fonction desquels la reconnaissance des gestes sera effectuée. Les vecteurs d'observation enregistrés sont : (a) les différences entre l'ordonné de chaque doigt et celle du centroïde ; (b) les abscisses des doigts et (c) les différences entre les abscisses des doigts adjacents. La reconnaissance statique se base uniquement sur les vecteurs du premier cas, tandis que la reconnaissance dynamique tient compte des trois cas.

La main du musicien prend souvent une posture inclinée, rendant la reconnaissance plus difficile. Suite à une recherche antérieure, des techniques d'invariance de rotation peuvent être appliquées à l'image. Etant donné que les doigtés sont repérés et identifiés, l'angle de la ligne qui résulte du centroïde et les coordonnées du majeur et de l'axe horizontal est calculé. Si cet angle dépasse un certain seuil, l'image pivote automatiquement à l'orientation inverse de l'inclinaison [13]. Cela permet une reconnaissance robuste même si la main est inclinée et assure l'indépendance des caractéristiques gestuelles, élément très important pour la partie entraînement des MMC. Pour des valeurs d'angle d'inclinaison inférieures au seuil, le système est capable de reconnaître le geste sans correction de la rotation.

5. MODELISATION, APPRENTISSAGE ET MAPPING

5.1. Modélisation stochastique et reconnaissance

L'extraction (*reconnaissance*) des doigtés, autrement dit reconnaissance statique, se met en œuvre en déterminant le seuil d'appui effectué sur une touche pour chaque doigt. Même dans le cas d'un « doigt caché », le doigté sera extrait dans les images suivantes.

La combinaison des doigtés forme un geste dit « pianistique ». Pour cela un dictionnaire des gestes ainsi qu'un alphabet des doigtés ont été créés. Les gestes, se projetant en mouvements musicaux, sont analysés à la fois harmoniquement et mélodiquement afin d'extraire leurs états structurels.

Pour l'instant, deux versions du système existent : (a) PianOrasis développé sur Matlab étant capable de reconnaître les gestes à la fois statiques et dynamiques des doigts en temps différé, et (b) PianOrasis:max, développé sur Max/MSP, étant capable de reconnaître les gestes statiques des doigts en temps réel.

Les valeurs continues des vecteurs d'observation, extraits par les séquences d'images, sont modélisées à l'aide des MMG, tandis que chaque geste est modélisé par les MMC [9, 10], offrant ainsi une certaine flexibilité à l'entraînement des modèles et permettant l'importation de vidéos de longueurs différentes ou de données manquantes.

Plus précisément, les MMC continus ont été choisis du fait (a) de la précision fournie dans la classification ; (b) qu'ils ne nécessitent pas de quantification des données ; (c) du petit nombre de données d'apprentissage pour les modèles. Le modèle du geste est évalué en estimant le maximum de vraisemblance (*similarité entre le geste effectué et les gestes modélisés*).

5.2. Correspondance de la musique au geste

Afin de faire correspondre le geste à la musique, on doit avant tout répondre à des questions comme « à quoi peut-on associer tel ou tel paramètre gestuel ou sonore ? » et « comment le faire ? ».

Dans le cadre de la méthodologie proposée, il s'agit de faire correspondre les caractéristiques des gestes des doigts de haut niveau aux paramètres acoustiques de bas niveau. Les caractéristiques extraites décrivant le geste du musicien sont automatiquement enregistrées dans un tableau. Dans l'hypothèse où les données gestuelles sont valables, les listes des données importées sont interprétées en tant que valeurs de contrôle. L'importation d'une nouvelle liste des données gestuelles alors correspond à un nouvel ensemble de valeurs dans l'espace sonore.

La méthodologie présentée ici repose sur une correspondance au niveau du temps (*temporal mapping*) au lieu d'une correspondance au niveau de l'espace (*spatial mapping*). Cela est lié au besoin de développer un système de mapping en temps réel. Ceci dit, une procédure temporelle de correspondance entre le geste et le profil temporel du son doit être déterminée [14]. La correspondance temporelle peut être considérée en tant qu'une procédure de synchronisation entre les paramètres gestuels d'entrée et les paramètres sonores de sortie. Le rythme d'exécution de geste peut être synchronisé à l'aide des procédures temporelles spécifiques. La correspondance temporelle peut ainsi être prolongée au contrôle d'une combinaison des profils temporels continus et d'événements discrets dans l'espace, en les synchronisant avec les caractéristiques du geste d'entrée.

Le principe de la méthodologie repose sur le fait que les gestes sont des procédures temporelles qui se caractérisent par des profils temporels. La notion de la correspondance temporelle est approchée en examinant les côtés temporels de la relation entre geste, son et structures musicales. C'est l'évolution temporelle des données et non pas leurs valeurs absolues qui devient un élément fondamental des systèmes musicaux d'interaction. Cette approche se base sur la modélisation des caractéristiques temporelles très spécifiques, lesquelles sont soit acquises lors de la procédure de l'apprentissage statistique, soit réglées manuellement.

6. SYSTEMES ET INTERFACES

La méthodologie a été mise en œuvre à l'aide de deux environnements de programmation différents. La première version du système a été faite sur Matlab (Figure 2) et la deuxième sur max/MSP (Figure 3).

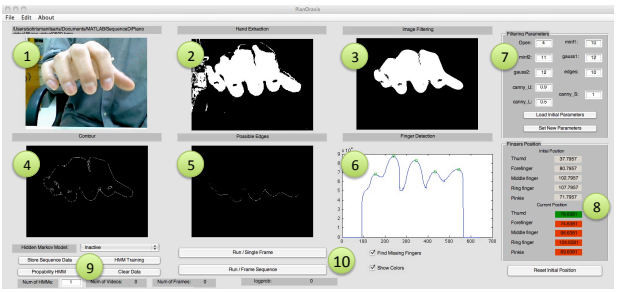


Figure 2. Interface du PianOrasis
(1) image initiale en RGB ; (2) application du modèle de la peau ; (3) application de la morphologie mathématique ; (4) extraction du contour de la main ; (5) arêtes probables correspondant aux bouts des doigts ; (6) localisation des bouts des doigts dans l'image et prédiction de la position des doigts cachés ; (7) paramétrage manuel des filtres et de la détection des contours ; (8) initialisation de la position de repos pour chaque doigt et détection des doigtés ; (9) entraînement des modèles gestuels à partir des vidéos ; (10) option de reconnaissance statique ou dynamique.

PianOrasis propose des fonctionnalités à la fois pour la reconnaissance statique (*reconnaissance des doigtés*) et la reconnaissance dynamique des gestes musicaux des doigts en temps différé (Figure 2). Le système, ainsi que son interface, ont été entièrement développés sous Matlab. Plusieurs toolbox ont été utilisées, telles que

«Image Acquisition» pour la capture des vidéos, «DIPimage» pour le traitement statistique de l'image et le «Kevin Murphy» pour la modélisation stochastique à l'aide des MMC et des MGM. PianOrasis assure différentes fonctions concernant l'importation et le suivi de traitement de la vidéo, le filtrage, l'entraînement et la reconnaissance.

PianOrasis:max est entièrement développé sous Max/MSP et propose actuellement des fonctionnalités de reconnaissance statique en temps réel (Figure 3). Max/MSP propose tout un ensemble de routines qui existent sous la forme des toolbox. La bibliothèque « Computer Vision for Jitter » de Jean-Marc Pelletier a été utilisé pour la segmentation de l'image mais en même temps d'autres routines appelées « objets extérieurs » ont été créées.

La mise en œuvre du module de mapping geste/son sera faite en utilisant le patch MnM. Il s'agit d'un toolbox de Max/MSP qui se base sur le patch FTM. Il est composé d'un groupe des unités qui procurent des algorithmes de l'algèbre linéaire de base, algorithmes de correspondance et des algorithmes de modélisation statique comme le PCA, GMMs et HMMs.

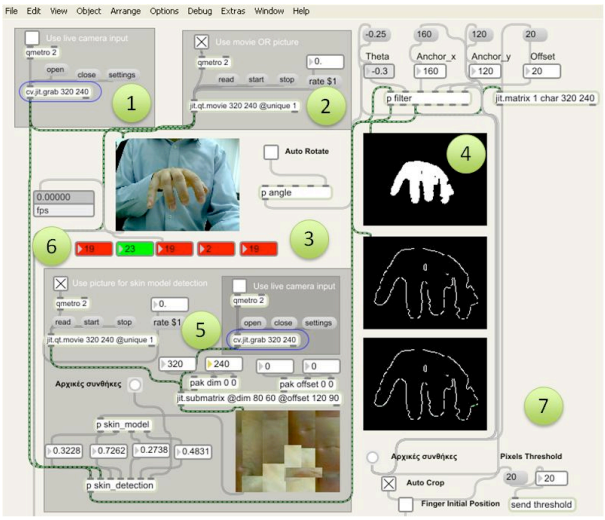


Figure 3. Interface du PianOrasis:max lorsque un doigté de l'annulaire est détecté
(1) Option de reconnaissance en temps réel ; (2) Option de reconnaissance en temps différé ; (3) vidéo initiale en RGB ; (4) application du modèle de la peau et de la morphologie mathématique ; (5) Option de paramétrage du modèle de la peau ; (6) Reconnaissance des doigtés ; (7) Option automatique du découpage de la RI et de la rotation de l'image

7. EVALUATION

Un premier scénario d'évaluation pour PianOrasis:max a été basé sur un geste d'arpège complet pour la gamme de Do ; un geste assez simple car les distances entre les doigts sont grandes et aucun doigt ne se cache. Nous avons demandé à un musicien d'effectuer ce geste en tempo 92 BMP. Une vidéo en 19 fps a été enregistrée pour environ 15 sec, soit 285 images au total. La main du musicien était pivotée par angle $\vartheta \in [15,20]$ et le taux de couverture de la peau sur l'image très basse. Le système a identifié avec succès tous les doigts pour la totalité des images de la vidéo ainsi que tous les appuis du musicien.

PianOrasis:max a été ensuite évalué sur huit vidéos (19 fps) d'une durée entre 15 et 20 sec et d'une couverture de la silhouette de la main sur l'image de l'ordre de 14% (Figure 4). Il s'agit des gestes de a) gamme ascendante, b) gamme descendante, c) arpège ascendant et d) arpège descendant toujours pour la gamme de Do. En comparant le temps moyen du traitement entre PianOrasis et PianOrasis:max pour les mêmes vidéos nous constatons une diminution de 31,7%. Cela est dû à la procédure du découpage de la RI aux limites de la silhouette de la main sur l'image binaire, procédure n'étant pas disponible à la version antérieure. Pourtant le taux de reconnaissance reste relativement constant entre les deux versions (*83,64% pour PianOrasis et 81,64% pour PianOrasis:max*)

Vidéo	Durée (sec)	Temps de traitement sans découpage (sec)	Temps de traitement avec découpage (sec)	Reconnaissance PianOrasis (%)	Reconnaissance PianOrasis:max (%)	Réduction du temps (%)
A	9	98.51	66.73	87.5	100	32.3
B	9	94.89	66.40	75	87.5	30
C	11	120.15	82.95	77.7	89	30.9
D	11	114.92	79.82	89	77.7	30.5
E	9	99	66.67	85.7	85.7	32.5
F	15	157.39	105.36	90.9	81.8	33.1
G	7	81.78	55.68	80	80	31.9
H	7	76.75	51.86	83.3	66.6	32.4

Figure 4. Trajectoires des bouts des doigts

Les trajectoires des cinq bouts des doigts du geste de la vidéo E apparaissent dans la figure 5. L'axe vertical représente la distance entre les coordonnées du bout du doigt et celles du centroïde tandis que l'axe horizontal représente les images de la séquence.

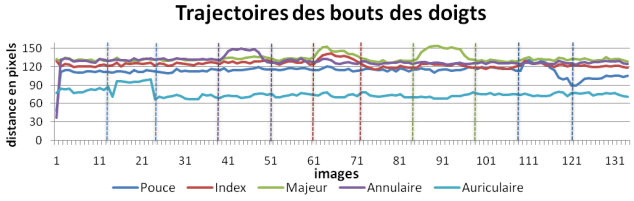


Figure 5. Trajectoires des bouts des doigts

Lorsque la distance entre le bout d'un doigt et le centroïde dépasse de 20 pixels le seuil, un doigté est enregistré. Les doigtés du geste de la figure 4 sont

indiqués par les lignes pointillées verticales de même couleur que la trajectoire du doigt. Il est facile de constater que le geste de cette vidéo correspond à une inversion majeur/index. Par ailleurs, le rythme de l'interprétation n'est pas constant et la position d'un doigt est souvent influencée par le geste des doigts adjacents.

8. CONCLUSION

Une méthodologie de vision par ordinateur pour la reconnaissance des gestes des doigts dans l'espace a été présentée. Actuellement, une nouvelle version du système PianOrasis:max, est disponible, en proposant de la reconnaissance des doigtés en temps réel. Des nouvelles techniques d'invariance d'échelle et de rotation ont été introduites, rendant ainsi le système moins dépendant de la distance entre la main et la caméra et des petites rotations de la main. Une méthodologie pour le contrôle gestuel du son pour chaque doigt indépendamment a été présentée. Elle sera mise en œuvre dans un très proche avenir. Le système a été évalué pour une gamme de gestes pianistiques effectués en espace sans aucune interférence entre les doits du musicien et l'ordinateur. Cela constitue une bonne base de recherche pour le développement d'un instrument de musique immatériel, un « espace interactif » pour la production du son, où le musicien ne sera pas limité par une interface spécifique. Un geste naturel effectué dans un environnement réel pourra déclencher un ensemble d'entités musicales, tels que le son d'un instrument musical ou même le segment d'une voix parlée. La connaissance des notions approfondies en musique ne constituera pas un préalable pour l'utilisation d'un tel instrument musical.

9. REFERENCES

[1] Manitsaris, S. *Vision par ordinateur pour la reconnaissance des gestes : analyse et modélisation stochastique du geste dans l'interaction musicale*, Thèse de Doctorat, Université de Macédoine, Thessalonique, Grèce, 2010.

[2] ViconPeak. *Vicon Motion Capture System*, Lake Forest, Ca, 2005.

[3] Palmer, C. & Pfordresher, P. Q. Frommy hand to yourear: the faces of meter in performance and perception. In C. Woods, G. Luck, R. Brochard, F. Seddon& J. A. Sloboda (Eds.) In *Proceedings of the 6th International Conference on Music Perception and Cognition*. Keele, UK: Keele University, 2000.

[4] Rasamimanana, N. & Bevilacqua., F. Effort-based analysis of bowing movements: evidence of anticipation effects. *The Journal of New Music Research*, 37(4):339 – 351, 2009.

[5] Coduys, T., Henry, C. and Cont, A. TOASTER and KROONDE: High-Resolution and High-Speed Real-time Sensor Interfaces, In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME-04)*, Hamamatsu, Japan, 2004.

[6] Bevilacqua, F., Zamborlin, B., Sypniewski, A., Schnell, N., Guédy, F., Rasamimanana, N. Continuous real time gesture following and recognition, *LNAI 5934*, pp. 73–84, 2010.

[7] N. Rasamimanana, F. Bevilacqua, N. Schnell, F. Guedy,, E. Come Maestracci, B. Zamborlin, JL. Frechin,U. Petrevski, « Modular Musical Objects Towards Embodied Control Of Digital Music », In *Proceedings of Tangible Embedded and Embodied Interaction*, 2011.

[8] S. Hadjidimitriou, A. Zacharakis, P. Doulgeris, K. Panoulas, L. Hadjileontiadis, and S. Panas, "Revealing action representation processes in audio perception using Fractal EEG Analysis: A Mirror Neuron System-Based Approach", *IEEE Transactions on Biomedical Engineering* , VOL. 58, NO. 4, pp. 1120-1129, 2011.

[9] Baum, L. An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. In *Proceedings of the Third Symposium on Inequalities*, New York, USA, 1972.

[10] Rabiner, L. R. A tutorial on hidden Markov models and selected applications in speech recognition». In *Proceedings of the IEEE*, 77(2), 257-285, 1989.

[11] Papamarkos, N., Strouthopoulos, C., &Andreadis, I. *Multithresholding of color and gray level images through a neural network technique", Image and Vision Computing*, vol. 18, 213-222, 2000.

[12] Manitsaris S. «Vision par ordinateur pour la reconnaissance des gestes musicaux des doigts : le système PianOrasis», *Revue Francophone de l'Informatique Musicale*, 1(1), MSH Paris Nord, 2011.

[13] Tsagaris, A., Manitsaris S., Dimitropoulos, K., Manitsaris, A. «Scale and rotation invariance for the recognition of finger musical gestures performed in space», In Proc. *Of the Third European Workshop on Visual Information Processing (EUVIP)*, Paris, France, 2011.

[14] Matsoukas, V., Manitsaris S., Manitsaris, A. «Finger gesture control of sound», In *Proc. Of the Third European Workshop on Visual Information Processing (EUVIP)*, Paris, France, 2011.

DE LA MUSIQUE MECANIQUE A LA « MECAMUSIQUE », UNE DEMARCHE BASEE SUR L'INFORMATIQUE MUSICALE DE COMMANDE

Jacques Rémus
Ipotam Mécamusique
info@mecamusique.com

RÉSUMÉ

L'histoire de la musique mécanique, depuis l'antiquité jusqu'à la deuxième guerre mondiale, procure un enseignement riche sur la volonté d'automatiser la musique, c'est-à-dire de la mémoriser et d'actionner des instruments mécaniques avec des énergies non humaines: le cylindre à picots en est un élément clé et le fonctionnement de ces systèmes, essentiellement à base d'orgues et de carillons, préfigurent celui des ordinateurs et de la robotique. La sculpture sonore ou Klangkunst [6] a développé une nouvelle dimension artistique, à la fois plastique et musicale, à partir de ces éléments. Après avoir décrit le parcours de l'auteur dans ce domaine (« Mécamusique »), il est présenté une analyse du processus de découplage entre le jeu instrumental et les actions sonores qui permet aux créateurs de développer leurs œuvres. La maîtrise des ordres, (écriture et enregistrement), des flux d'énergies (transmissions et actionneurs) et la conception de la lutherie et de la musique a pour centre l'informatique musicale de commande qui permet de jouer avec les spécificités de l'écriture musicale pour la robotique comme les décalages temporels (offsets) dus à la mécanique. Des exemples d'éléments pour structurer les logiciels de commandes sont indiqués.

1. INTRODUCTION

« Für Augen und Ohhren » [2] à l'Académie des Beaux-Arts à Berlin, fut une exposition (venue ensuite à Paris : « Ecouter par les yeux » en 1980 [23] qui mit en avant le « Klangkunst »[6] Ce mot se traduit mal par sculpture sonore ou arts plastiques du son, c'est-à-dire une pratique artistique mêlant la sculpture cinétique et la musique, une pratique ou un minimum de technologie est nécessaire : le Klangkunst s'inspire de la musique mécanique traditionnelle et de la robotique mais c'est avant tout une nouvelle forme d'art hybride, à la fois visuel et musical [14].

2. LE DESIR D'AUTOMATISER LA PRODUCTION DE LA MUSIQUE : RAPPELS HISTORIQUES

La création d'instruments de musique capables de jouer automatiquement est un vieux mythe de l'humanité : les orgues éoliens qui nous sont parvenus par exemple des traditions des habitants de l'Ile de Guadalcanal[1] sont basés sur l'énergie et l'aléatoire du vent. Leur lutherie permet de faire ressortir des sons de flutes réalisées avec des bambous dans lesquels des trous ou lèvres sont taillés. Ils fonctionnent « automatiquement », mais le fil de la partition reste incontrôlé.

2.1. L'Antiquité

L'antiquité gréco-romaine nous révèle une volonté de créer des machines automatiques pour produire de la musique instrumentale. L'eau est la source d'énergie, les notions d'horloge, d'orgues et de régulateurs hydrauliques sont décrites[2] mais les hydraules semblables à celui qu'aurait inventé Ctésibios à Alexandrie ne sont sans doute pas des automates, il n'y a pas trace de système qui mémorise le déroulement temporel d'une partition. Néanmoins, d'après les écrits [3] il est fort probable que des mécanismes de programmation ont existé.

2.2. Du Moyen-Age à la Renaissance

C'est au 9ième siècle, à Bagdad que l'on trouve trace de la description d'un véritable automate musical[4]. L'énergie vient d'une chute d'eau avec une roue à aube. Celle-ci entraine la production d'air comprimée (vent) par des soufflets et la rotation d'un cylindre garni de

[1] collectage de Hugo Zemp :Orgue éolien, ghau kilori) Iles Salomon, Musique de Guadalcanal, Ocora France, C 580049.

[2] retrouvés par exemple en 1931 à Aquinum (Hongrie) et daté de l'an 228 AC ou en 1993 à Dion en Macédoine et daté du premier siècle BC.

[3] Philon de Byzance (300 av.JC), Apollonius (200 ans av.JC), Heron d'Alexandrie (1 siècle ap.JC) qui a écrit Pneumatica" sur des fontaines et des automates musicaux..

[4] ``Al alt allati tuzamiru binafsiha`` (l`instrument qui souffle de lui-même).

picots qui marquent les notes de la partition (comme dans les boîtes à musiques actuelles), la musique est jouée par une flûte et des tiges déclenchées par les picots actionnent l'ouverture des trous des notes de la flûte. Tous les éléments d'une machine musicale sont assemblés : énergie, ordres mémorisés et actionneurs pour aboutir au son instrumental et à la musique automatique. Ces développements ont été liés à ceux des horloges et à des réalisations d'oiseaux chanteurs. On attribue à Ibn Ismail Ibn al-Razzaz Al-Jazari la création de remarquables de robots humanoïdes musiciens en 1206. Mais ces inventions n'ont pas laissé de traces.

Sans entrer dans les détails historiques qui ont marqués des échanges entre le monde arabe, Byzance avec ses orgues et serinettes, la Chine avec ses carillons à programme et les royaumes occidentaux, il faut remarquer que plusieurs développements concourent à la création de machines musicales de plus en plus perfectionnées. En effet l'orgue avec son clavier et ses commandes mécaniques va devenir un instrument qui découple le jeu instrumental de l'instrument, tandis qu'au 14ième siècle les carillons des églises et beffrois non seulement utilisent le même genre de mécanisme, avec des marteaux, mais utilisent une programmation par cylindres pointés qui permet de jouer de véritables partitions de manière automatique et programmée. De même les « Jacquemarts »[5] allient l'art du carillon relié au marquage des heures de l'horloge avec la notion de personnages animés, sculptures cinétiques dont le mouvement provoque le son. L'énergie vient de contrepoids et les cylindres, ancêtres de nos séquenceurs, se perfectionnent. Ils ont des picots (c'est-à-dire des ordres de notes mémorisées car piquées sur le cylindre) interchangeables qui plus tard porteront, par décalage face aux commandes d'actionneurs, plusieurs partitions (comme les pistes d'un séquenceur qui auraient plusieurs musiques en parallèle, mais n'en jouerait qu'une à la fois). Bien sûr la longueur de chaque partition est limitée au tour du cylindre. Léonard de Vinci dessine en 1493 [5] et [22] des machines à percussion tractées par homme ou par animal qui, en avançant, font tourner des axes à programme (Figure 1) qui animent les baguettes.

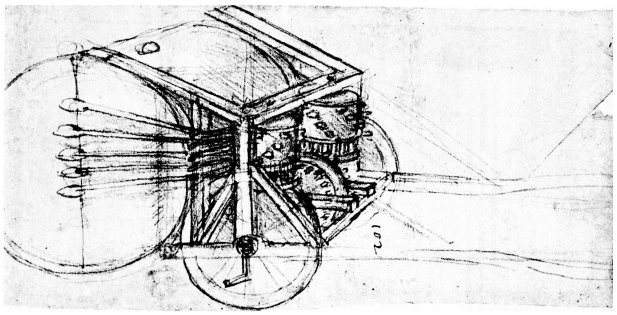


Figure 1. Léonard de Vinci: tambour à 10 baguettes et double axe à programme, tracté (CA_306_va)

[5] personnage sculpté en bois ou en métal, qui indique les heures en frappant une cloche avec un marteau. L'un des plus anciens en France est celui de Dijon.(1383)

2.3. Du 16ième siècle jusqu'à l'ère industrielle

Il faut noter les recherches, écrits et dessins de Salomon de Caus [4] , de Robert Fludd [8], de Marin Mersenne [12], d'Athanasius Kircher [10] au 17ième siècle, et de Dom Bedos [1] et de Joseph Engramelle [7] au 18ième siècle qui décrivent ou imaginent des machineries musicales extraordinaires, et la manière de les programmer. Elles sont en général destinées à des jardins d'agrément. L'énergie y est surtout hydraulique et le cylindre à picots omniprésent [9] et [17].

Salomon de Caus décrit et dessine des machines avec de très gros cylindres à programme (Figure 2).

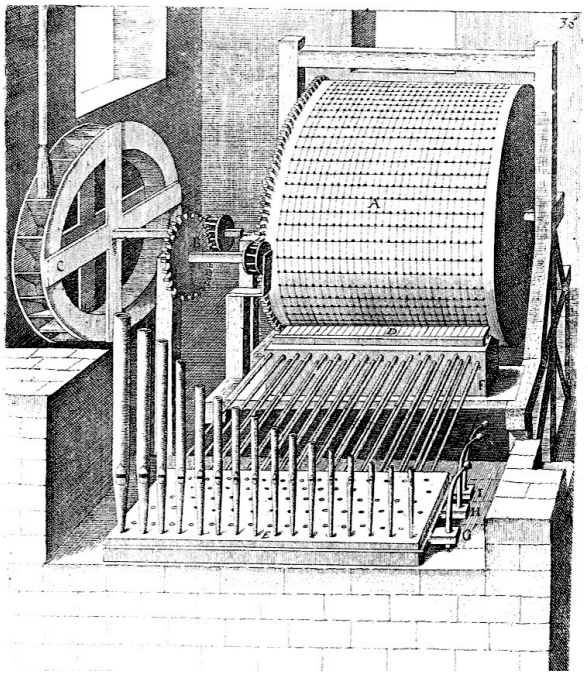


Figure 2. Orgue automatique dessiné par Salomon de Caus, l'énergie qui fait avancer la partition est hydraulique.

Kircher décrit des carillons tel qu'ils fonctionent encore dans les beffrois du Nord de la France et en Belgique (Figure 3). Il décrit aussi, avec des dessins qui ont illustré depuis bon nombre de livres ou d'affiches de Festivals des systèmes complexes destinés à faire fonctionner des grottes "magiques" avec orgues automatiques et oiseaux chanteurs. Les jardins de Tivoli en Italie gardent trace de telles installations.

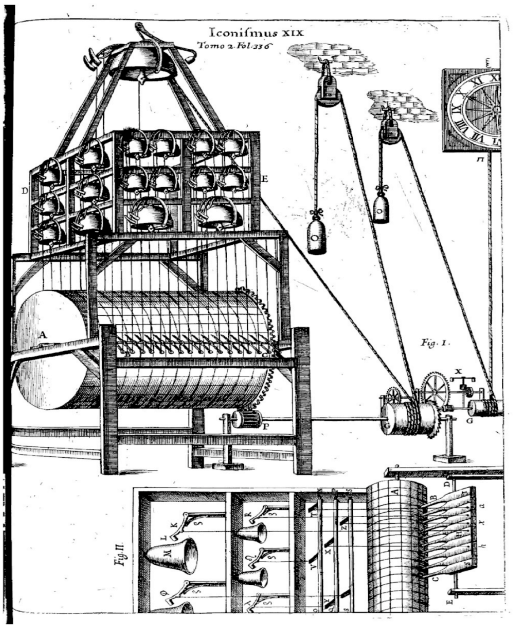


Figure 3. Carillon automatique dessiné par A. Kircher

Au 18ième siècle, la course à la précision des horloges pour la navigation, en Angleterre, en Hollande, en France vont de pair avec le développement d'automates de plus en plus perfectionnés : apparaissent de nombreuses horloges qui comportent des mouvements avec flûtes ou carillons, des orgues à cylindre miniatures destinés à entraîner des oiseaux chanteurs (Serinettes), des orgues à cylindre plus importants pour les salons et les églises, des automates musiciens très précis que réalisent des artisans comme Bidermann (1540-1624), Vaucanson (1709-1772), ou Pierre Jacquet-Droz (1721-1780). Des compositeurs comme G. F. Händel, Joseph Haydn, W.A. Mozart puis L. von Beethoven ont écrit pour ces instruments. A la même époque se créent aussi des boîtes à musique à lames vibrantes, dont le mouvement peut se loger dans de tous petits objets [3].

Mais c'est au 19ième siècle que deux phénomènes vont transformer la musique mécanique [21]:

- le remplacement des cylindres à picots ou à taquets par l'utilisation de lecteurs de cartons et rouleaux perforés, issus de l'industrie textile[6] : ils donnent la possibilité d'une partition sans limite de temps.
- et les utilisations d'énergies industrielles comme la vapeur (sur les orgues Calliope par exemple), l'air comprimé (qui permet la musique interfacée sur les orgues romantiques avec les machines Barker) et enfin l'électricité.

2.4. Du 19ième au 21ième siècle, apogée puis déclin de la musique mécanique, naissance de l'informatique musicale et de la musique robotisée

L'apogée de la musique mécanique, à la fin du 19ième siècle, jusqu'aux années 30 du 20ième siècle se termine avec l'apparition de l'enregistrement audio sur cylindre puis sur disque, du haut-parleur et de l'amplification électrique.

La musique mécanique, parfois un peu rugueuse, sans nuance d'intensité ou de rythme (sauf pour le Pianola) et dont on a oublié l'importance et le succès a permis l'accès facile à la musique à un large public dans les salles de danse, les cafés, les fêtes foraines, la rue et dans les salons bourgeois. Elle a initié la consommation de masse de la musique tout en retirant du travail aux musiciens. Le phonographe et la radio l'ont remplacé. Les orchestrions, Pianola, Accordéo-Jazz, Phonolist-Violina, Mills-Violano, Magic-Organa, Wurlister, orgues de foires géants vont s'éteindre comme les dinosaures !

Seuls quelques compositeurs comme Igor Stravinsky, Alfredo Casella, Paul Hindemith ou Colon Nancarrow utiliseront les ressources de ces machines qui ont massivement disparues avec la deuxième guerre mondiale. Mais les principes de l'informatique musicale de commande sont là : ordres avec mémoires, énergie et action sur de la lutherie, temps réel ou lecture de données enregistrées. Ce sont aussi les concepts de base des ordinateurs et de la robotique.

2.5. Naissance d'une nouvelle pratique artistique : la sculpture sonore

Influencés par les mouvements artistiques comme le futurisme de Luigi Russolo [18] et les débris de la musique mécanique, naissent à partir du milieu du 20ième siècle des créations d'artistes qui cherchent à faire voir et faire entendre des phénomènes sonores générés à distance, automatiquement ou préenregistrés. La musique est générée par des actionneurs qui se réfèrent parfois à la lutherie traditionnelle, sur les bases des cordes, vents et percussions diverses [20] et parfois à des détournement d'objets industriels devenant tous des « corps sonores » au sens de Pierre Schaeffer [19].

Le propos se situe entre la musique, art du temps et les arts plastiques (cinétiques), art de l'espace et parfois du spectacle, art de la scène (Figure 4).

[6]Les métiers inventés par Joseph-Marie Jacquard à Lyon en 1801.

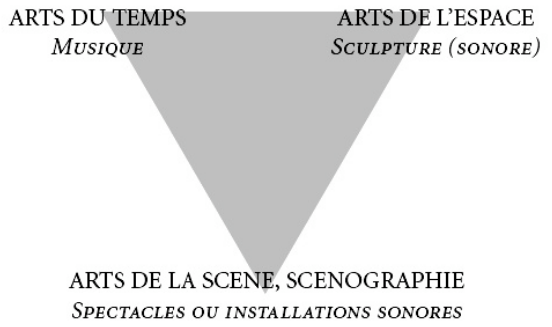


Figure 4. Triangle symbolique dans lequel se situent les réalisations du domaine du « Klangkunst » ou sculpture sonore [14]

Les démarches dans ces domaines sont aussi nombreuses que les créateurs, mais elles ont pour point commun de privilégier le mouvement qui produit le son avec ses caractéristiques acoustiques et spatiales. On peut citer pêle-mêle des auteurs comme Harry Bertoia, Nam June Paik, Harry Partch, Jean Tinguely, Takis, Frédéric Lejunter, les frères Baschet, Rebecca Horn, Pierre Bastien, Michel Moglia et beaucoup d'autres.

Très vite la programmation avec l'électromécanique, la mise en mémoire d'ordre sur bandes optiques ou bandes magnétiques devient l'outil nécessaire pour piloter ces créations.

Lorsque le Midi, langage de communications d'informations (ordres) dont la structure a été pensée musicalement, apparaît au début des années 80 avec la généralisation de l'ordinateur individuel, c'est une révolution pour le monde de la musique, la scène et les studios.

Il paraît incroyable que ce langage, dont on connaît pourtant les défauts et dont l'environnement de transmission a considérablement évolué, soit inchangé depuis 30 ans, alors que tous les langages informatiques et les protocoles de transmissions se transforment, apparaissent, disparaissent.

Fait d'ordres et non de sons il est l'essence des partitions informatiques et des transmissions en particulier entre synthétiseurs, samplers et ordinateurs.

C'est bien sûr le protocole que les auteurs de sculptures sonores et d'installations de musique robotisées vont majoritairement adopter.

Parmi les nombreux auteurs qui se sont lancés dans cette direction en utilisant l'informatique musicale de commande et le Midi, il faut citer des artistes comme Trimpin (USA), Gordon Monaham (Canada et Allemagne), Chritoph Schläger (Allemagne et Pays-Bas), Nobumichi Tosa (*Maywa Danki*, Japon), Martin Riches (Grande Bretagne et Allemagne), Gotfried Willem Raes (*Logos*, Belgique), Eric Singer (USA), Peter Bosh et Simone Simons (Pays-Bas, Espagne), Chico MacMurtrie (USA), Rogers Troy (USA), etc. [6].

3. PARCOURS PERSONNEL AVEC LA COMPAGNIE « IPOTAM MECAMUSIQUE », DIFFUSEUR DES CREATIONS DE SCULPTURES SONORES ET MACHINES MUSICALES

J'ai été formé, lors de premières études, à l'informatique des grosses machines utilisant des systèmes d'acquisition à cartes perforées (IBM 80 colonnes) et demandant de grandes pièces climatisées pour faire péniblement beaucoup moins que ce que fait allégrement un smartphone.

Puis j'aI gouté à la lourdeur de la programmation de Music V à l'école de Pierre Schaeffer (Groupe de Recherche Musicale)

Cependant, j'étais attiré par l'informatique et je voulais lui faire jouer au moins le rôle des cylindres à picots de la musique mécanique, car j'étais désireux de composer avec des sons physiquement présents et toujours disponibles (au contraire des instrumentistes !). J'avais rencontré la musique libérée du geste humain de Colon Nancarow mais je n'en étais qu'à mes premiers essais de machines télécommandées avec des programmateurs électromécaniques ou simplement mécaniques (disques à picots et chaînes à « événements »).

Cependant l'apparition de petits ordinateurs d'expérimentation comme le ZX81 de Sinclair, a permis, avec un programme en langage Basic, d'automatiser partiellement le spectacle Bombyx dés 1982-83 [16].

L'apparition du langage Midi changea les perspectives et après quelques créations audio avec un ordinateur Yamaha équipé en Midi, de norme MSX, j'ai enfin pu travailler la musique sur ordinateur, et donc la robotique grâce à des démultiplexeurs et des cartes de puissance spécialement conçus.

L'interface Roland MPU 401 a été la première interface utilisée, avec un séquenceur canadien «T.N.S.» et un clône taïwanais d'ordinateur IBM PC en 1986 : le spectacle du Double Quatuor à Cordes est ainsi né au Canada. en 1987, puis Concertomatique N°1 en 1990.

L'étape suivante fut l'initiation à Max à l'Ircam avec Miller Puckette. La première version (1992) de la Camera Musicale [13] fut expérimentée grâce à un algorithme Max pilotant les données de l'interface Manorine développée par Sylvain Aubin (1982).

La création de Concertomatique N°2 donna lieu au développement des embryons des outils actuels dont les caractéristiques sont :

- 1) la commande directe et contrôle de tout actionneur en Midi, DMX ou RS 232 (ou 422), parfois via OSC ;
- 2) l'assemblage de sous-programmes permettant un jeu direct sur l'ensemble des données robotiques en jouant simplement d'un clavier Midi, avec toutes les possibilités de nuances que permet la robotique qui lui répond ;

- 3) la commande par interfaces non classiques : capteurs divers, caméras, microphones (par pitch-riders), etc.;
- 4) la possibilité de pré-programmations permettant de multiplier les possibilités musicales à partir de gestes simples ;
- 5) l'enregistrement, l'arrangement et la composition sur un séquenceur Midi performant.

Les interfaces destinées au jeune public comme les joystick, les « Wii », se sont rajoutées pour les commandes des ordres.

Les installations sonores ou spectacles comme le Carillon des Zic-Phones, les Grilles, la Sonnette, Léon et le Chant des Mains, le Carillon N°3, l'Ensemble des Machines Musicales, l'Orgabulles, Les Thermophones à Bascules, les Bascules à Percussions, les Thermophones 2 sont toutes régies sur ces principes.

4. ANALYSE DU PROCESSUS DE DECOUPLAGE ENTRE JEU INSTRUMENTAL ET ACTION SONORE : ORDRES , ENERGIE, ACTION, LES OUTILS A LA DISPOSITION DES CREATEURS

L'évolution de la technique de l'orgue est un exemple facile pour comprendre le phénomène de découplage : les premiers hydraules avaient des ouvertures de tuyaux directes avec des sortes de tirettes. L'ouverture de l'arrivée de l'air (vent) dans un tuyau pouvait donc être progressive et l'organiste devait pouvoir nuancer l'attaque et même l'intensité des notes. Les premières touches de clavier, larges et ouvrant de simples clapets devaient permettre les mêmes nuances. Le jeu du musicien était couplé à l'instrument, seul le souffle du vent était extérieur à lui.

L'orgue baroque introduit des transmissions directes mais parfois éloignées vers l'ouverture des passages de vent dans les tuyaux. Même si l'organiste peut nuancer légèrement l'attaque des notes, nous sommes en présence d'une machine télécommandée en « tout ou rien ».

L'orgue romantique introduit l'amplification des ordres du clavier par l'intermédiaire d'une machine pneumatique (inventée par Charles Spackman Barker en 1837). Le découplage est alors complet et les systèmes électro-pneumatiques puis électriques des orgues modernes n'ont rien changé à cette caractéristique. Ce découplage a donc, de plus, fait perdre toute possibilité expressive, mais c'est un phénomène indépendant. L'orgue liturgique contemporain peut être joué avec une console distante (claviers et pédalier) et peut même jouer tout seul, piloté avec un ordinateur.

4.1. Ordres, énergie, lutherie écriture et enregistrement

4.1.1. Ordres, écriture et enregistrement.

Les supports d'écriture ont été historiquement des axes à ergots (automates mécaniques éoliens en Asie du sud-est), puis des cylindres, des disques, des cartons perforés, des rouleaux de papier perforés et maintenant des logiciels séquenceurs. L'écriture y est strictement proportionnelle au temps (Figure 5).

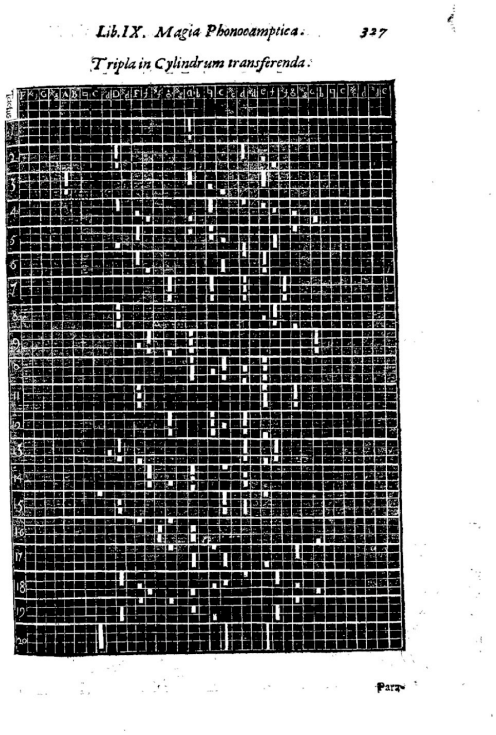


Figure 5. Déroulé de la partition d'un cylindre pour orgue montrant la position et la durée des notes (Athanasius Kircher, [10]).

Un petit ouvrage intitulé la « tonotechnie » a été écrit par Joseph Engramelle en 1775 [7] pour proposer la maÎtrise et les astuces de cette notation (Figure 6).

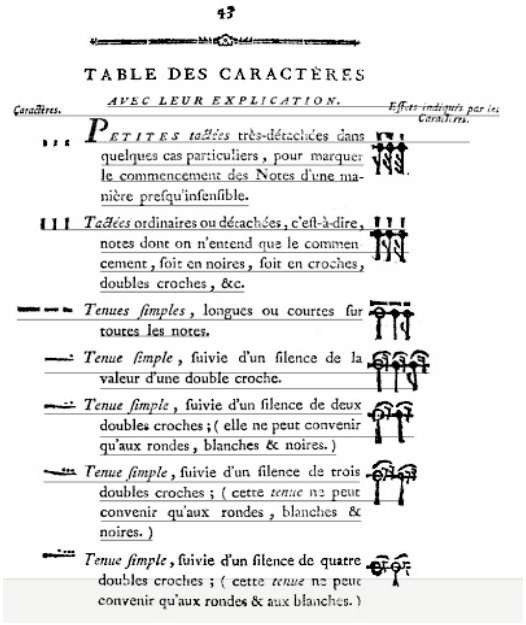


Figure 6. Indications pour la notation des cylindres ou tonotechnie. Joseph Engramelle [7].

La notation respectant les nuances rythmiques du jeu humain ne sont pas faciles et les partitions de musique mécanique sont souvent marquées par une grande raideur. Les Pianola ont cependant pu être munis d'enregistreur sur papier qui ont conservé non seulement l'enregistrement rythmique mais des jeux de pédales de pianistes célèbres au début du 20ième siècle. Le même problème se pose avec un séquenceur (Figure 7) et les « noteurs » qui perforent les cartons des orgues de Barbarie font rarement autre chose que de recopier une partition note à note, comme on peut le faire sur un séquenceur. La souplesse rythmique de musiques de jazz a cependant pu être maitrisée par des noteurs comme Pierre Charial, mais la plupart travaillent maintenant avec des perforatrices pilotées en Midi !

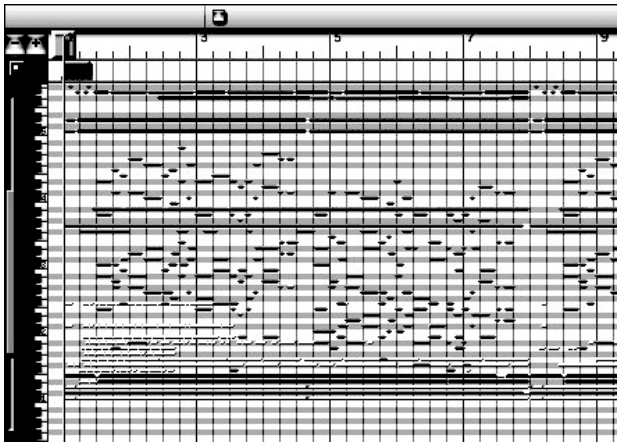


Figure 7. Notation en « rouleau de piano » (pianoroll) sur séquenceur d'une pièce pour Concertomatique N°2 : ici sont superposées les ordres de notes musicales et les ordres de moteurs, de pression, de lumières, etc..

4.1.2. Energies, transmissions et actionneurs.

La transformation à vue des énergies de tout ordre en son est un élément capital de la lutherie mécanisée ou robotisée: les sources sont actuellement plus facilement électriques (réseau, accumulateurs ou capteurs solaires), mais il peut toujours y avoir mis en oeuvre de nombreux types d'énergie directe ou intermédiaire qui sont parfois spectaculaires et font partie de l'oeuvre: la chute ou la pression de l'eau, le mouvement des soufflets, l'air comprimé pour le vent de l'orgue, le poids du mécanisme horloger, l'air en pression négative des pianos et orchestrions mécaniques (boîtes à touches, flûte de pan et transmissions), la chaleur du feu ou des capteurs solaires, la vapeur des calliopes (orgues à vapeur), l'air comprimé des orgues à Klaxon, l'air à très haute pression [15], etc.

La transmission de ces énergies en mouvement par des mécanismes en bois, en métal, par des tuyaux de fluide sous pression ou par des engrenages peut faire partie de l'œuvre, mais elle est souvent remplacée par des connexions électriques en parallèle ou en données mises en série par langage informatique ; la connectique et les interfaces font alors directement partie du système géré par l'informatique. Le démultiplexeur est alors le passage obligé entre la commande et l'action.

Les actionneurs ne peuvent fonctionner que si les ordres ont été amplifiés par l'énergie apportée : le vent des machines Barker et maintenant l'électricité dans les cartes de puissance. Les actionneurs peuvent alors être « tout ce qui bouge » : électro-aimants, électrovannes, moteurs, motoréducteurs, moteurs pas à pas, pompes, vérins, résistances et tout ce que l'industrie développe pour ses chaînes de montage, ses robots aussi bien que ce qui sert dans les véhicules ou les avions!

4.1.3. Lutherie et musique.

La lutherie ou facture instrumentale reste, avec l'écriture musicale et son support informatique, l'élément le plus important de ces systèmes. C'est dans ces deux domaines que l'imagination et la création sont rois. Au sens musicologique [20] les aérophones, les cordophones, les membraphones et les idiophones forment les quatre grandes familles de l'acoustique musicale des œuvres. Les développements industriels ou leurs propres recherches ont apporté aux artistes des nouveaux champs d'investigation, mais, même par exemple avec les sons itératifs devenant des fréquences sonores ou la thermo-acoustique, il est possible de ramener les vibrations acoustiques de ces instruments aux notions de colonnes d'air, de plaques, de membranes, de cordes ou de masses vibrantes[11].

Il faut noter que les cordes demandent des systèmes d'amplifications ou de résonnances importants et qu'en général l'obtention de sons puissants dans les graves demande beaucoup d'énergie et... d'imagination.

La musique peut s'écrire en amont mais les particularités de ces systèmes font que son arrangement in situ est souvent générateur de modifications qui

deviennent écriture à part entière tellement le contexte peut-être singulier ou plein d'artefacts.

4.2. Outils pour créateurs : importance de l'informatique musicale de commande

4.2.1. Points forts et points faibles.

La musique mécanique traditionnelle est synonyme de musique raide, métronomique, souvent à timbre grinçant et éloignée du jeu et de l'interprétation humaine.

L'utilisation de l'informatique de commande et d'actionneurs développés par l'industrie permet de palier à ces inconvénients, en introduisant trois changements fondamentaux:

1) la souplesse rythmique (enregistrements Midi et modifications sur séquenceurs) ;

2) la nuance et la sensibilité (vélocité des notes et actionneurs à variations) ;

3) l'équilibre des basses avec les actionneurs et la lutherie robotisée bien adaptés à cette préoccupation.

Ce sont cependant des notions coûteuses en temps et en argent et certaines démarches en font l'économie car la primauté peut-être donné à l'art cinétique, au mouvement des mécanismes et actionneurs et à l'aspect scénographique et plastique de l'œuvre.

Mais il reste que ces installations ont plusieurs points forts qui les caractérisent:

1) Le son est réel, avec tous ses constituants en particulier les transitoires d'attaque et ses aspérités qui lui donnent une présence et une qualité que ne peuvent égaler les systèmes de haut-parleurs les plus perfectionnés ;

2) la spatialisation est réelle: la caractéristique de la musique robotisée a ceci de commun avec la musique instrumentale, c'est que, en écoute non amplifiée, la spatialisation est « parfaite » , mais bien sûr pour des auditoires limités. C'est aussi pourquoi les installations de sculptures sonores et de musique robotisée, au contraire de la musique mécanique traditionnelle, ont tendance à privilégier l'occupation de l'espace et de proposer une immersion dans le son dont les sources multiples peuvent être par exemple des notes dispersées ;

3) la possibilité de commandes par interfaces multiples : les ordres peuvent provenir de musiciens jouant directement ou de variations aléatoires ou contrôlées données par des capteurs ;

4) la possibilité de jouer des musique affranchies des limites du geste humain et donc d'explorer des univers sonores et des créations musicales inouïes.

4.2.2. Spécificités de l'écriture musicale pour robotique : les offsets.

Le travail d'enregistrement, d'écriture ou d'arrangement nécessite la présence de tout le système : en effet il peut y avoir une grosse différence entre la simulation sur sons synthétiques et le rendu réel sur machines. Et de toute façon, il y a de nombreux ordres robotiques qu'il faut ajouter pour faire fonctionner l'ensemble.

Par ailleurs, le décalage temporel dû à l'inertie des systèmes mécaniques, pneumatiques, etc. est une chose qu'il faut absolument connaître et maîtriser.

Par exemple, un vérin pneumatique actionnant des percussions va avoir une inertie temporelle de plusieurs dizaines de millisecondes par rapport à l'ouverture d'un clapet de flûte qui sera comparativement instantanée. Si de plus le vérin pneumatique est à pression variable, un son faible aura une inertie, donc un « retard », beaucoup plus grande (centaines de millisecondes) qu'un son fort. Les moteurs ou électroaimants de percussions à tension variable auront les mêmes caractéristiques.

Donc, à tous les actionneurs seront affectés des offsets qui, s'ils sont variables, pourront être calculés et de toute façon réglés à l'oreille et corrigés directement sur le séquenceur.

Pour enregistrer, il faut disposer d'un développement sur un logiciel de type Max et d'un séquenceur permettant de bien travailler avec le Midi. Une installation commode et sûre pour créer les pièces musicales est indiquée sur la figure [9]. Elle demande deux ordinateurs, mais elle offre une plus grande souplesse et une sécurité qu'avec un seul ordinateur qui fait tourner séquenceur et Max en même temps car il se trouve que des datas parfois disparaissent dans les connexions internes de soft à soft et ici on a l'assurance d'être dans la bonne configuration pour jouer.

L'écriture pour les Thermophones par exemple est un vrai poème : leur inertie est de l'ordre de plusieurs dizaines de secondes et elle varie selon la température de l'air dans les tuyaux, donc en fonction du nombre de fois ou ils sont joués et en fonction du temps entre l'arrêt de la précédente note et le démarrage prévu de la suivante sur le même tuyau.

4.2.3. Nature des enregistrements.

Il faut remarquer que deux types d'enregistrements sont possibles :

-le premier est celui qui garde les actions jouées par exemple sur un clavier ;

-le second garde ce qui part vers les machines.

Pour être rejoué, le premier, plus léger, nécessitera tous les algorithmes (ou sous-patches) (Figure 8) qui permettent aux actionneurs d'être pilotés, le second, beaucoup plus riches en datas, pourra être utilisé avec n'importe quel lecteur de midifile directement sur les machines (Figure 9).

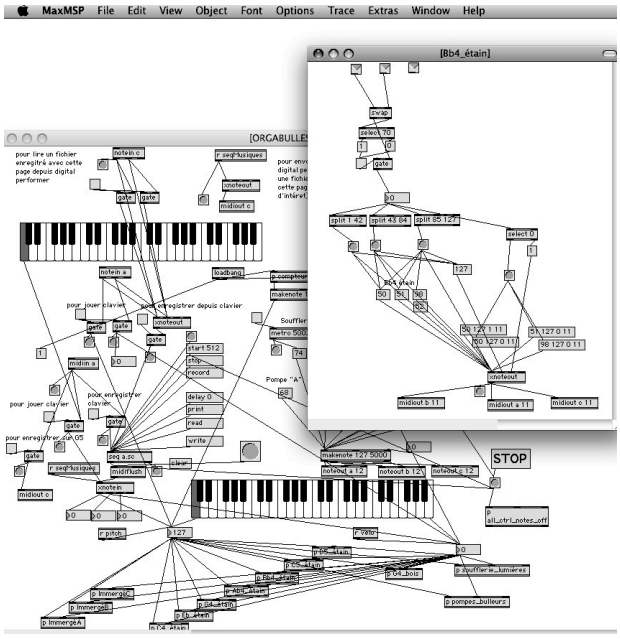


Figure 8. Exemple de cœur de transformation des ordres du clavier en ordres robotiques par algorithmes adaptés (sous-patches)

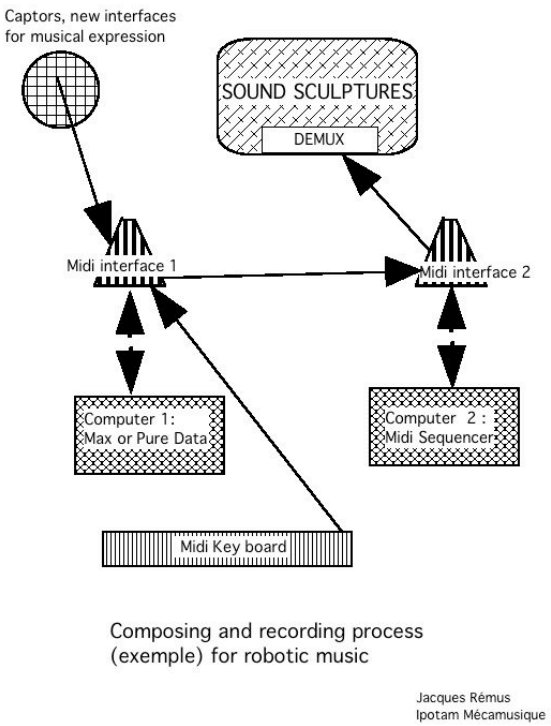


Figure 9. Exemple de mode d'enregistrement permettant de garder les données des ordres en amont ou en aval du traitement pour actionner la robotique.

4.2.4. Structure des logiciels.

Une fois les enregistrements et arrangements faits, un logiciel d'exécution est bâti dans un programme comme Max ou Pure Data. Il peut être composé d'une dizaine de modules indépendants qui se retrouvent sur tous les types d'installation et qui permet en particulier de coupler et de synchroniser des installations de musique robotique indépendantes (Figure 10):

- d'un répertoire donnant accès aux midi-files enregistrées ;
- d'un lecteur en général unique (seq ou detonate par exemple sur Max), ;
- d'un sélectionneur de midifile automatique, jouant les fichiers les uns après les autres, ou au hasard ;
- d'un accès pour le public lui permettant de choisir une musique par l'intermédiaire d'une interface physique ;
- d'un autre accès pour le public qui permet de jouer en direct des sculptures sonores via une interface (clavier, Camera Musicale, etc.) ;
- d'une section maintenance qui permet d'accéder à chaque fonction robotique avec des sous-patches de fonctionnements partiels synchronisés, d'une section de commandes pour les interfaces temps réel (clavier, caméra Musicale, etc…) ;
- d'une section horloge qui permet de déclencher le système à des moments précis ou de faire jouer plus fort certaines pièces à certaines heures ;
- d'une section salle ou spectacle qui permet d'agir par exemples sur les lumières qui ne sont pas automatisées, etc. (Figure 11) ;
- d'un enregistrement horo-datés des noms des pièces jouées pour les déclarations de droits d'auteur…

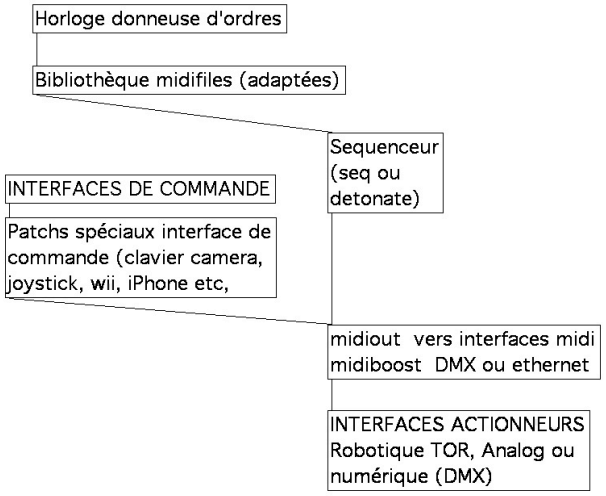


Figure 10. Schéma simplifié de la structure d'un logiciel de commande de sculptures sonores.

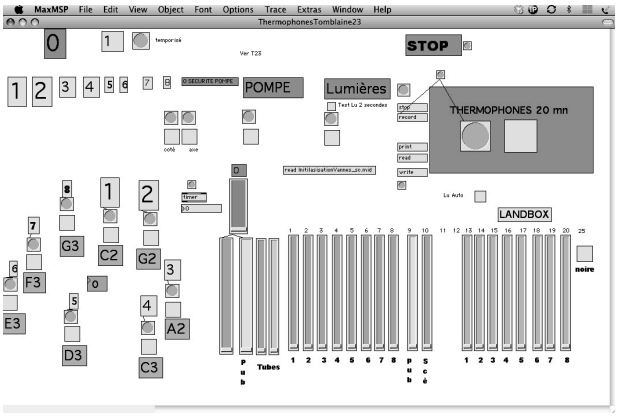


Figure 11. Exemple d'un développement simple permettant de tester tous les éléments d'une installation, de lancer un spectacle automatique et de gérer la salle indépendamment (régie lumière).

5. CONCLUSION

La démarche que les allemands ont nommé le "Klangkunst" [6] est une voie artistique qui de par sa nature mêle les arts du temps et les arts de l'espace. C'est un art impur au sens où l'a décrié Goethe.

Elle est très éloignée, dans l'esprit, de la musique mécanique traditionnelle, mais elle partage avec elle la volonté de faire vivre la musique jouée « magiquement » et la nécessité d'utiliser, pour fonctionner, les technologies de pointe de son époque. L'informatique musicale de commande pilotant l'électronique et les actionneurs de la robotique en sont les caractéristiques actuels.

Des matériaux nouveaux, des nouvelles possibilités logicielles et électroniques, des actionneurs nouveaux se développent sans cesse et les artistes se feront un plaisir de les détourner pour leurs créations.

Artistes et chercheurs ont maintenant des champs de travail qui s'ouvrent sur des développements comme l'interactivité avec intelligence artificielle pour des jeux entre machines et musiciens, sur la création d'installations à radio-commande très éclatées dans l'espace pour envahir des lieux insolites ou des lieux publics, l'utilisation d'énergies autonomes venant par exemple de capteurs solaires, etc. L'informatique musicale de commande en sera bien sûr l'outil clé.

6. REFERENCES

[1] Bedos de Celles , (Dom) F., *L'art du facteur d'orgues*. Paris 1766-78

[2] Block R., Dombois L., Hertling N., Volkman B., *Für Augen und Ohren*, Akademie der Künste, Berlin 1980.

[3] Bowers Q. D., *Encyclopedia of automatic musical instruments: Cylinder music boxes, disc music boxes, piano players and player pianos... Incl. a dictionary of automatic musical instrument terms*. Vestal, N. Y., The Vestal Press, 1988.

[4] De Caus S., *Les Raisons des forces mouvantes, avec diverses machines tant utiles que plaisantes*, Jan Norton, Francfort, 1615; Paris, Jérôme Drouart, 1624.

[5] Da Vinci, L., *in Codices Madrid – Traztato di estatica y mechanica*, Italiano, 1493.

[6] De La Motte-Haber H., *Klangkunst,Tönende Objekte und klingende Räume*,Laaber-Verlag, 1999.

[7] Engramelle M. D. J., *La tonotechnie ou l'art de noter les cylindres et tout ce qui est susceptible de notage dans les instrumens de concerts méchaniques*, Paris 1775.

[8] Fludd R. (Robertus de Fluctibus), *Utriusque cosmi majoris et minoris historia*, Oppenheim 1617

[9] Haspels Jan Jacob Leonard, *Instruments their mechanics and their music* 1580-1820, Utrecht,1987.

[10] Kircher A., *Musurgia universalis, sive ars magna consoni et dissoni* , Rome, 1650.

[11] Leipp É., *Acoustique et musique*, Masson, Paris,1984.

[12] Marsenne M., *L'Harmonie universelle*, Paris, 1636

[13] Rémus J., "Non haptic control of music by video analysis of hand movements : 14 years of experience with the «Caméra Musicale»", Proceedings of the 2006 International Conference on New Interfaces for Musical Expression (NIME06), Paris, France, 2006.

[14] Rémus J., *La sculpture sonore, pratique artistique en recherche de définition*, In Musiques Arts Technologies, pour une approche critique, Music arts technologies, toward a critical approach", (Barbanti R., Lynch E, Pardo, C., Solomos, M), L'Harmattan, Paris, France, 2004.

[15] Rémus J., "The music of ringed pipes", Proceedings of the International Symposium on Musical Acoustics, Paris, France, 1995.

[16] Rémus J., "Mécamusique" , PUCK n°6 ,Musiques en Mouvement, edited by Institut International de la Marionnette, 1993.

[17] Rouillé P., Musée d'instruments de musique mécanique, Gamm, Paris,1987.

[18] Russolo L., *L'arte dei rumori*, 1916, réed/trad., Lausanne, L'age d'homme, 1975.

[19] Schaeffer P., *Traité des objets musicaux*, Paris, Ed du Seuil, 1966

[20] Schaeffner A., *Origine des instruments de musique, Introduction ethnologique à l'histoire de la musique instrumentale*, Mouton & Co et la Maison des Sciences de l'Homme, 1968.

[21] Stanley S. (Ed.): *Musical Box*. *The New Grove Dictionary of Music and Musicians*, MacMillan. 1980.

[22] Winternitz, E., *Leonardo da Vinci as a musician*, Yale University Press, 1982.

[23] Catalogue d'exposition, *Ecouter par les yeux, Objets et environnements sonores*, ARC Musée d'Art Moderne de la Ville de Paris,Paris,1980

« NAMUDA STUDIES » : DOOPLER RADAR-BASED GESTURE RECOGNITION FOR THE CONTROL OF A MUSICAL ROBOT ORCHESTRA

Godfried-Willem Raes
Ghent University College,
School of Arts & Logos
god@logosfoundation.org

ABSTRACT

This paper describes the application layer for gesture recognition using Doppler-based hardware systems, described in full detail in two papers referenced in the bibliography: one on the hardware used [1] and another on the gesture recognition software engine [2]. The hardware implements a fully non-contact gesture acquisition system based on reflected waves from the naked skin. The recognition software is largely based on fuzzy logic for classification of gesture properties.

Being capable of recognizing a defined set of about twelve expressive gestures in a piece of software is of little significance if an application layer fails. Namuda dance technique [3] requires a mutual adaptation of the performer and the software parameters. In order to make the study of Namuda dance possible, we have designed a series of études in which each single gesture prototype can be practised. Since visual feedback to the performer is very problematic in the context of performance, for it greatly hinders freedom of movement and is by nature too slow, we have opted for auditory display. The robot orchestra [4] as we have designed and built it, makes a good platform for such auditory display, particularly since the sounds are not virtual (loudspeakers) but real acoustic sounds emanating from real physical objects. In fact just about any musical instrument can be seen as an example for auditory display as it by its very nature truthfully converts a certain subset of fine motor skills and gestures into sound. The gestures underlying music practice may very well constitute a basis for the embodiment underlying the intelligibility of music. [5] The motor skills and gestures entailed by playing traditional musical instruments are obviously instrumental in nature. They are dictated by the mechanical construction of the instrument. Therefore, as an extension of the body, an instrument can, at most, be a good prosthesis. By removing the necessity of a physical object, the body becomes the instrument. But this in no way removes the need for motor skill and gestural control.

1. NAMUDA ETUDES

The scheme followed for each étude is always the same: starting with the default parameter settings in the software, the performer has to practice each gesture prototype until the recognition is optimum, as can be judged from the response of the robots. The default parameters are not arbitrary, but have been determined as a result of hundreds of measurement sessions with about twenty different subjects, male and female. Each gesture prototype is mapped to a different subset of responding robots. In this respect, the study of Namuda gestures is quite similar to the study of any musical instrument. A certain level of fine motor control has to be developed in the player. Only once that level has been reached can the recognition software be modified by changing the parameters slightly. One would never buy a new and better violin for a child every time it makes a handling and playing mistake. Only once it knows the basics reasonably well should buying a better instrument become an option. Fortunately, in the case of the invisible instrument, we do not have to buy a new instrument but we can improve the software and adapt it to the player. This last possibility opens a whole new perspective for future developments in instrument building.

1.1. Speedup

To study steady accelerating movements, we made a mapping to the x, y and z vectors for the oboe (<Ob> robot), the cornet (<Korn> robot) and the saxophone (<Autosax> robot) respectively. The speedup property strength is mapped to the pitch the instruments will sound whereas the sound level is controlled by the value of the speed parameter. It is a good exercise to try to have the instruments play as long as possible by stretching the time over which the property can remain in a set state. As soon as the property is set in all three vectors, the large stainless steel shells that make up <Llor> will come into play. The sensitivity is set at a much lower level for this to happen than is the case for the vectors taken separately.

1.2. Slowdown

The slowdown property can obviously only be set when the starting gesture is in movement already. Thus the property can never be triggered from rest. The étude maps all three vectors to pitches in the <Piperola> and

<Bourdonola> robots. These are flue pipe organs. The pitch depends on the value of the slowdown, not on the property strength. When the property is set in all three vectors, the lights on the piperola robot will flash.

1.3. Expanding

To trigger this property a movement is required whereby the amount of moving body surface gradually is increased. The property is associated with growth, explosion, enlargement and can be triggered from a standstill. The x-vector is mapped to our <Klung> robot, an automated Indonesian anklung with brass chimes. Pitch selection is mapped to property strength whereas volume is correlated to the value of the property. The y-vector similarly is mapped to <Simba>, a cymbal playing robot. The z-vector is mapped on <Springer>, a somewhat hybrid robot combining shakers, very large springs mounted on resonators as well as a big motor-driven siren. The selection of the elements playing in this robot is mapped to the duration of the property in the z-vector.

When all three vectors are set, the lights on the <Simba> robot will come up.

1.4. Shrinking

Being the other side of the dipole to the previous property, the gestural associations can also be formulated as imploding, diminishing, getting smaller. Clearly the property presupposes movement to start from. The mapping is as follows: x-vector to <Xy>, a quarter tone xylophone robot; y-vector to <Tubi>, a quarter-tone robot made with thick aluminium tubes; z-vector to <Vibi>, an automated vibraphone. For all three vectors, property strength is mapped to the pitch and property value on the sound level. When the property is triggered in all three vectors the lights on <Xy> will flash.

1.5. Steady

In order to trigger this property it is required that the amount of body surface in motion remain constant within the time frame of measurement. The mapping for this property is on our quarter-tone organ robot <Qt>. When the property is set in all three vectors, the blue lights on the robot will come on. In this mapping the amount of body surface remaining constant determines the pitches of the notes. The attack velocity of the notes is controlled – in a rather subtle way – by the property strength.

1.6. Fixspeed

This property is set as soon as the detected speed of a movement stays reasonably constant within a time frame of 500 ms. It is pretty difficult at first to trigger this property more than just accidentally. The reason is not only due to our control, but also in part due to the cosine factors on the Doppler frequency shifts. The latter can be much improved and even cancelled out by keeping the angle of the movement axis and the sight of the vectorial transducer constant. The mapping of all three vectors here is on one of our smallest robots: <Toypi>, an automated toy piano. When all three vectors have the property set, the lights on the little robot come up.

1.7. Collision

Since determination of this gesture property is based on acceleration followed by a sudden stop, it implies a well-defined sudden stop in the gesture. Rebounding movements should be avoided as they can result in false or double triggering. To make the étude convincing, we mapped the output to nothing but percussion, the collision-based instrument family par excellence. The étude should be performed using all possible parts of the body: not only arms and legs, but also the head, the entire torso, the feet, the elbows and even the belly if musculature allows it... The x-vector is mapped to the drums in <Troms> (a set of drums) and the <Snar> robot, an automated snare drum. The y-vector is mapped to the the cowbells that make up the <Vacca> robot. The z-vector is mapped to the set of woodblocks in the <Thunderwood> robot as well as to the thin metal sheets in the <Psch> robot. When collision is detected in all three vectors, the cymbal in the <Troms> robot will play. If collisions are detected but they are below the sensitivity level set in the software, the white lights on the robots will flash.

1.8. Theatrical Collision

As this prototype is defined as a decelerating movement ending in a stop and accelerating again – avoidance of real collision – it tends to be set more easily over relatively larger time spans. In our étude we mapped the gesture to our <Puff> robot, a percussive organ-like instrument tuned in quarter tones.

1.9. Smooth (roundness)

1.10. Edgy

These gesture prototypes can be practised together. The edgy property strength is mapped to the piano notes, played by our player piano robot. The smooth property is mapped to the pitches of our quarter-tone organ robot, <Qt>. The attack velocity is controlled by the momentary value of the vectorial moving body surface. The property is set based on an analysis of the spectrum of the Doppler signals. Edgy movement with many sudden changes causes an overweight of higher partials in the spectrum. The sound volume from <Qt> is made to be a (slow) function of the value of the low side of the power spectrum. It had to react slowly

because it steers the compressor on the organ. Those motors, due to their inertia, cannot change speed suddenly. The recognition quality was rated as excellent by all performers we have subjected to this étude so far.

1.11. Jump (airborneness)

The airborne gesture property is set when the performer jumps and no part of the body is at rest. False triggering can occur on large stepping movements though. The larger or higher the jump the stronger the property will be defined. For this étude the x, y and z-vectors are respectively mapped to <Heli> (a helicon robot) <Bono> (a trombone robot) and <So> (the sousaphone robot). When the property is defined for all three vectors, both castanet-playing robots will join in. The sound volume is mapped on the amount of body mass involved whereas the pitch will be proportional to the property strength.

When all three vectors have the airborne property set, certainty is nearly 100%. If only two vectors trigger the property, certainty is still higher than 90%. Most often it is the Z-vector that fails to trigger, which is easily explainable by the fact that the transducer for that vector is suspended above the performer. In order to be certain that the Z-vector also triggers the property, the jump must also include a positional displacement. Needless to add, this étude is quite exhausting to perform.

1.12. Periodic

This is the least well-functioning property in our set and it definitely needs further improvement. Response time is too slow and false triggers do occur regularly. The mapping to the drums in our <Troms> robot allows practice and evaluation. We have also developed software allowing performers to synchronize midi or audio file playback with gestural input. Within strict limitations such as no sudden tempo changes, avoidance of rhythmical complexities (doublings and triplets) it can even be got to work. Further research is being done based on cepstrum analysis.

1.13. Freeze

To practice this 'non-gesture', we applied an inverse mapping, whereby sound will be heard as long as the freeze property is not triggered. The robot used in this mapping is <Bourdonola>, a low register open organ pipes with a string-like sound. The étude is very fundamental as it lets the performers experience the very high sensitivity level of the system.

Parallel to these recognition-based gesture properties, the implementation also offers a full set of very direct mappings of movement parameters on sound output:

- moving body surface: The most intuitive mapping for this parameter seems to be to sound volume or density.
- speed of movement: The most intuitive mapping for this parameter seems to be to pitch.
- spectral shape of the movement: The most intuitive mapping for this complex parameter seems to be to harmony.
- acceleration of the movement: The most intuitive mapping for this parameter seems to be to percussive triggers.

Of course there is nothing mandatory in the way the mappings of gestural prototypes have been laid out in these études. It is pretty easy to devise mappings more suitable for use out of the context of our own robot orchestra. The simplest alternative implementations consist of mappings on any kind of MIDI synth or sampler. However mapping the data from our gesture recognition system to real time audio streams (as we did in our 'Songbook' in 1995, based on human voice sound modulation via gesture) is an even better alternative.

2. EXTENDING THE TIME FRAME

The gesture prototypes practised in the études reflect a gestural microscale in the time domain. Their validity may be as short as 7ms and can for most properties seldom exceed 2 seconds. The only gesture properties that can persist over longer time spans are freeze, periodic, edgy, smooth, fluent and fixspeed. Some can, by their nature, only be defined over very short time intervals: airborne and collision. These gesture prototypes are to be compared to what phonemes are in spoken language, although they already carry more meaning than their linguistic counterparts. Meaning here being understood as embodied meaning.

By following assignment and persistence of the gesture prototypes over longer time spans, say between 500 ms and 5 seconds, it becomes possible to assign expressive meanings to gestural utterances. Here we enter the level of words and short sentences, to continue using linguistic terminology as a metaphor. When we ask a performer to make gentle movements in order to express sympathy, then the statistical frequency of a limited subset of gesture properties will go up significantly. When we ask for aggression, the distribution will be completely different.

It is on this level of time scale that quite a few of our gestural prototypes may be correlated to the classifications of 'effort' set up by Rudolf Laban in his text 'Effort ; Rhythmic Control of Man's activities in Work and Play'. This text was written well before 1950 and only got published as an appendix to the book mentioned in the bibliography of our paper. (Laban,

1980). As far as we can judge, his classification and notation proposals are difficult to hold in a more generalized context of a theory of expressive gesture. Clearly none of Laban's analysis are based on any other objective measurement than visual observation and the dancers' own experience of effort.

It would be an interesting research project to find out how comparable such distributions for a limited set of sentic forms [6] are amongst a large set of different dancers and musicians. Unfortunately this is beyond the scope of our own mission. In part, such research is being done now by our colleague Dr. Jin Hyun Kim.

3. RELATION TO OTHER DANCE PRACTICES

Although as soon as we gained some insight in the potential for dance offered by our technology – in the mid nineteen-seventies – we carried out artistic experiments with dancers trained in classical ballet as well as modern dance, we quickly found out that such an approach to dance was unsuitable to work well with this technology. Classical dance forms concentrate on elegance and – in general – avoid collision and a sense of mass. Position in space and visual aspects are very dominant. Immediately alternative dance practices came into consideration. In the first place butoh dance, an avant-garde dance practice with its roots in Japan, where we also came in contact with it (through Tari Ito). Thus we got in contact with dancers such as Min Tanaka, Tadashi Endo, Emilie De Vlam and later on Lazara Rosell Albear ... This has led to quite a considerable list of collaborative performances. However, butoh is only vaguely defined from a technical dance point of view. Its non-avoidance of roughness, its nakedness [7] and its concentration on bodily expression, leaving out any distracting props and requisites, formed the strongest points of attraction. Only in some forms of contact improvisation did we find other links, but in this dance form we ran into problems with our technology, which is not capable of distinguishing the movements of more than a single moving body at the same time. As far as couple dances go, we have also investigated tango (and the related milonga) quite a bit, in part also because we happen to be a tanguero ourselves. In this type of dance the problem of the two inseparable bodies poses less of a problem since movements are always very well coordinated. Acrobatic tango is of particular interest for the gestures used are very well defined.

Other dance forms we have experimented with include pole dance, flamenco and break dance. For the first dance style, we even developed a special set of microwave radar sensors that can be mounted at the top of the pole. Unfortunately we found out that the professionals in this dance form have little if any affinity with the art forms we are interested in ourselves, which are after all still quite experimental. As for break dance, we are forced to admit that despite the fascination we have for the virtuosity in movement that can be found in the genre, it seems to be strongly bound to a certain age group that we have long since left behind us...

4. INTERACTIVE COMPOSITION AND CHOREOGRAPHY

It will be clear that the mastering of Namuda opens wide perspectives for the development of real time interactive composition with a strong theatrical component. Over the 35 years that we have been developing the system, many hundreds of performances have been staged.

The entire Namuda system including the invisible instrument is open for use by other composers and performers. Scientists interested in research into human gesture are also invited to explore the possibilities of the system. Other composers that have made use of it so far are Kristof Lauwers and Yvan Vander Sanden. The system has also been investigated by Hans Roels, Troy Rogers, Jin Hyun Kim, Dirk Moelants and others. Many applications have been developed on other platforms than our own GMT-programming environment. To facilitate this, we have designed a limited gesture sensor with built-in signal processing using a PIC micro-controller. This 'Picradar' sensor, as we have baptized it, makes use of microwave radar (9.35 GHz) and outputs its data following the midi protocol.

There is still a lot of work left to be done on improvements to the hardware and recognition software as well as in terms of its artistic implementations. An open invitation.

5. ACKNOWLEDGEMENTS

Thanks to my dance and music collaborators who helped me perform the many necessary experiments and measurements using their bodies: Moniek Darge, Angela Rawlings, Helen White, Dominica Eyckmans, Lazara Rosell Albear, Zam Ebale, Marian De Schryver, Nicoletta Brachini, Emilie De Vlam, Tadashi Endo, Nan Ping, Jin Hyun Kim, Marjolijn Zwakman and many others. Thanks also to my collaborator Kristof Lauwers for helping out with code development, data analysis and debugging.

6. ENDNOTES

[1] The complete hardware description of the system can be found at: http://www.logosfoundation.org/ii/Holosound_ii2010.pdf. The background to our experimental instrument-building projects is largely taken from "An Invisible Instrument", Godfried-Willem Raes, 1997: http://logosfoundation.org/g_texts/invisins.html, which deals in greater depth with the philosophy behind the design.

[2] The software gesture recognition layer is decribed in full in this paper: http://www.logosfoundation.org/ii/Namuda_GestureRecognition.pdf

[3] Namuda is a word of our own casting and stands short for 'Naked Music Dance'.

[4] Detailed information on the robot orchestra and the robots constituting it can be found at http://www.logosfoundation.org/instrum_god/manual.html

[5] These matters were discussed in depth in my texts on the invisible instrument: Raes,1993 ,1994 and 1999.

[6] The notion of sentic forms was introduced by Dr. Manfred Clynes, with whom we had many conversations and discussions at the time we visited him in Sydney when we were demonstrating our invisible instrument at the Sydney Conservatory. References to his publications on the subject can be found in the bibliography section of this paper.

[7] We wrote an essay on nakedness some time ago, after realizing that even nowadays there are still people around that seem to have difficulty in coping with this utmost human property... The text can be read at: http://logosfoundation.org/g_texts/naked.html.

7. RÉFÉRENCES

- BECKMANN, Petr & SPIZZICHINO, Andre "The Scattering of Electromagnetic Waves from Rough Surfaces", Pergamon Press, Oxford, 1963
- CLYNES, Manfred "Sentics, the touch of the emotions" , Anchor Books (NY, 1978)
- DROITCOUR, Amy Diane "Non contact measurement of hearth and respiration rates with a single-chip microwave Doppler radar", Stanford PhD thesis, June 2006.
- KRAMER, Gregory (ed.) "Auditory Display", Addison-Wesley Publishing Company, (Reading MA, 1994)
- LABAN, Rudolf "The mastery of movement", Macdonald & Evans Ltd (Plymouth, 1980)
- LEMAN, Marc "Embodied Music Cognition and Mediation Technology", (Cambridge, 2008)
- RAES, Godfried-Willem "Een onzichtbaar muziekinstrument" (Gent, 1993, Ph.D. thesis)
- RAES, Godfried-Willem "An Invisible Instrument (1994): http://logosfoundation.org/g_texts/invisins.html
- RAES, Godfried-Willem "Gesture controlled virtual musical instruments" (Ghent, 1999): http://logosfoundation.org/ii/gesture-instrument.html
- RAES, Godfried-Willem "Quadrada" (Ghent, 2003): http://logosfoundation.org/ii/quadrada.html
- RAES, Godfried-Willem "PicRadar" (Ghent, 2004-2005): http://logosfoundation.org/ii/picradar.html
- RAES, Godfried-Willem "Distance sensing" (Ghent, 2007): http://logosfoundation.org/ii/distance_sensing.html
- RAES, Godfried-Willem "Naked" (Ghent, 2008): http://logosfoundation.org/g_texts/naked.html
- RAES, Godfried-Willem "Microwave Gesture Sensing" (Ghent, 2009):
- http://logosfoundation.org/ii/dopplerFMradar.html
- RAES, Godfried-Willem "Holosound ii2010, a doppler sonar based 3-D gesture measurement system" (Ghent, 2011): http://logosfoundation.org/ii/holosound_ii2010.pdf
- RAES, Godfried-Willem "Namuda Studies" (Ghent, 2012): http://www.logosfoundation.org/scores_gwr/Namuda_Links/namuda_studies.html
- RAES, Godfried-Willem "Namuda gesture recognition for musical practice" (Ghent, 2011): http://www.logosfoundation.org/ii/Namuda_GestureRecognition.pdf
- ROADS, Curtis "The computer music tutorial", (MIT press, 1996)
- ROELS, Jetty(ed.) "Images of corporeality, traces of Butoh in Flanders", VCIT, (Ghent, 2002)
- SEIFERT, Uwe, HYUN KIM, Jin (eds) "Paradoxes of Interactivity", (Bielefeld, 2008)

NEURAL NETWORKS FOR MUSICAL CHORDS RECOGNITION

J. Osmalskyj, J-J. Embrechts, S. Piérard, M. Van Droogenbroeck
INTELSIG Laboratory, University of Liège, Departement EECS
josmalsky@ulg.ac.be

ABSTRACT

In this paper, we consider the challenging problem of music recognition and present an effective machine learning based method using a feed-forward neural network for chord recognition. The method uses the known feature vector for automatic chord recognition called the Pitch Class Profile (PCP). Although the PCP vector only provides attributes corresponding to 12 semi-tone values, we show that it is adequate for chord recognition.

Part of our work also relates to the design of a database of chords. Our database is primarily designed for chords typical of Western Europe music. In particular, we have built a large dataset filled with recorded guitar chords under different acquisition conditions (instruments, microphones, etc), but also with samples obtained with other instruments. Our experiments establish a twofold result: (1) the PCP is well suited for describing chords in a machine learning context, and (2) the algorithm is also capable to recognize chords played with other instruments, even unknown from the training phase.

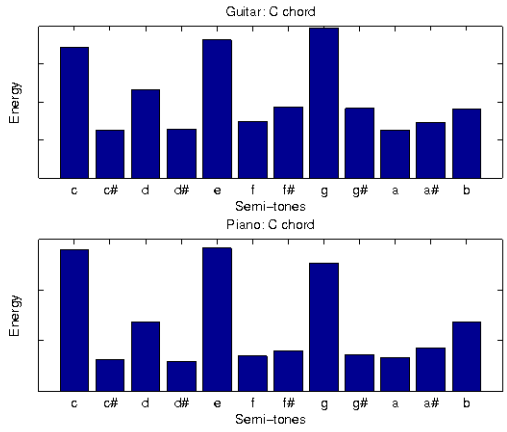


Figure 1. This paper shows that PCP features are well suited for representing chords, regardless of the used instrument. To illustrate the robustness with respect to the instruments, the upper and lower drawings provide the representation of a C chord played with a guitar and with a piano respectively. The three main peaks are the same.

1. INTRODUCTION

With the widespread availability of digital music over the Internet, it has become impossible to process that huge amount of data manually; even organizing a personal collection of music samples is challenging. Therefore automatic tools have a role to play. Music Information Retrieval (MIR) is an interdisciplinary science whose goal is to extend information retrieval into non textual-only areas. The aim of MIR is to describe multiple aspects related to the content of music. Some applications of MIR include music transcription, music classification [1], playlist generation [8, 20], and music recognition [3].

Traditionally, music is annotated with text information provided by the cover. This text information is adequate to characterize lyrics automatically but totally inappropriate to describe music content. Clearly, an approach based on text lacks flexibility. In addition, researchers are also interested in extending audio information retrieval using a more human natural interaction, for example, humming a song [3], tapping rhythm, or playing an instrument.

The first compulsory step of a retrieval system able to process music is the characterization of music. Several techniques are available but the probably best known to musicians is that of chords. A chord can be defined as *a set of simultaneous tones* [16]. This definition might be appropriate for a human but, from a classification point of view, it appears to be inappropriate because there are many variations (due to the instruments, noise, recording conditions, etc) even for a unique chord.

From a technical point of view, we show in this paper that Pitch Class Profile (PCP) features [9] are suitable candidates as they have a small sensitivity to instrument change (see Figure 1).

The PCP is a compact representation of the frequency content of a sound expressed as the relative proportion of energy with respect to the 12 notes of a standard chromatic scale. Notes are defined on a logarithmic frequency scale, and energy is expressed in natural units. Most of the studied PCP features are sensitive to the harmonics depending on the musical instrument, and other parameters such as dynamic, attack and sustain. To take these influences into consideration, we propose, in this paper, a novel approach using machine learning techniques.

But a good descriptor does not suffice. In the paper, we establish that a naive application of the definition of chords to classify PCPs fails to provide good results. In addition, we also prove that once this diversity is correctly handled by machine learning methods, chords form an adequate description for recognizing musics. During our work, we used machine learning techniques for chords

recognition. However, such algorithms usually need a labeled database in order to learn a classification model. As, to our knowledge, there is no such database publicly available. Therefore, we have created a new large dataset, which is publicly available.

The remainder of this paper is organized as follows. Section 2 reviews the related work in the field of chord characterization and classification. Section 3 provides a brief reminder of the PCP feature vector and details why a learning algorithm is preferable to a nearest neighbor approach. Section 4 describes our new database, its design, and its content. Section 5 presents the results of experiments, and Section 6 concludes the paper.

2. RELATED WORK

Chroma features, also known as Pitch Class Profiles (PCP), have been used as front end to chord and songs recognition systems from audio recorded queries. In particular, it has been demonstrated that chroma features are well suited for cover songs identification systems [5, 6, 7, 11, 14, 15, 17].

PCP features are good mid-level features which provide a more reliable and straightforward representation of songs than melody. In [18], Serra describes the PCP features as derived from the energy found within a given frequency range in short-time spectral representations of audio signals. This energy is usually collapsed into a 12-bin octave independent histogram representing the relative intensity of each of the 12 semitones of an equal-tempered chromatic scale.

The original PCP was introduced by Fujishima [9] in 1999. In this PCP, the intensities of all frequency bins within the boundaries of a semitone are summed-up and the semitones in octave distance are added-up to pitch classes, resulting in a 12-bin PCP vector. Variations of this vector include 24-bin and 36-bin vectors, resulting in more precise features. Fujishima used his PCP vector to perform pattern matching using binary chord type templates (*i.e.* ideal PCP representations as shown in Figure 2).

Lee [16] introduced a new feature vector called the Enhanced Pitch Class Profile (EPCP) for automatic chord recognition from the raw audio. To this end, he first obtained the Harmonic Product Spectrum (HPS) from the constant Q transform (CQT) of the input signal and then he applied an algorithm to that HPS for computing a 12-dimensional enhanced pitch class profile. The CQT has geometrically spaced center frequencies which can be dimensioned so that they correspond to musical notes. It is thus an interesting pre-processing step for music computing.

Gomez and Herrera [10] proposed a system that automatically extracts, from audio recordings, tonal metadata such as chord, key, scale and cadence information. In their work, they computed a vector of low-level instantaneous features: the HPCP (Harmonic Pitch Class Profile) vector. It is based on the intensity of each pitch mapped to a single octave, which corresponds to Fujishima's PCP.

Harte [12] also proposed a method using the CQT for chord recognition. In addition, he added a tuning algorithm which is able to deal with variations in instrument tuning.

Sheh and Ellis [19] proposed a statistical learning method for chord segmentation and recognition, where they used Hidden Markov Models (HMMs) trained by the Expectation Maximization (EM) algorithm, and treated chords labels as hidden values within the EM framework.

Most of the aforementioned work on chords recognition does not make use of machine learning techniques, but rather uses signal processing techniques in order to obtain the best possible 12-bin PCP vectors, and then perform pattern matching. However, it is very difficult to obtain a perfect 12-bin PCP vector which highlights only the main notes of a chord. Indeed, each instrument brings new harmonics, and the dynamic of the musician, among other parameters, adds noise to the PCP. For this reason, we propose a system based on machine learning techniques, whose goal is to learn a suitable model encapsulating all these parameters.

However, no real labelled chords database seems to be publicly available (to our knowledge) to build such a model. In this work, we propose a database, and we consider the use of real chords samples to train a more accurate chord recognition system. Since our goal is to use our system for music recognition, we need fast algorithms, which are necessary to deal with huge databases of songs. Therefore, we chosed to use the original PCP vector because it is fast and involves few pre-processing steps.

3. CHROMA FEATURES

3.1. Principle

The most commonly used descriptor for chord identification has been the Pitch Class Profile (PCP). A chord is composed of a set of tones regardless of their heights, and therefore a PCP vector seems to be an ideal feature to represent a musical chord.

There are some variations to obtain a 12-bin PCP, but its computation usually follows the same steps. First the algorithm transforms a fragment of the input sound to a discrete Fourier transform (DFT) spectrum $X(.)$. Then the algorithm derives the PCP from $X(.)$. Let $PCP^*(p)$ be a vector defined for $p = 0,\ 1,\ \ldots,\ 11$ as

$$PCP^*(p) = \sum_l ||X(l)||^2\, \delta(M(l),\, p) \qquad (1)$$

where $\delta(\cdot, \cdot)$ denotes Kronecker's delta. $M(l)$ is defined as

$$M(l) = \begin{cases} -1 & l = 0 \\ round(12\, log_2((f_s \cdot \frac{l}{N})/f_{ref}))\, mod\, 12 & l = 1,\ \ldots,\ \frac{N}{2} - 1 \end{cases}$$

where f_{ref} is the reference frequency falling into $PCP^*(0)$, N the number of bins in the DFT of the input signal, and f_s is the sampling frequency. For example, for a standard scale starting with a C, the reference frequency is $130.80\, Hz$.

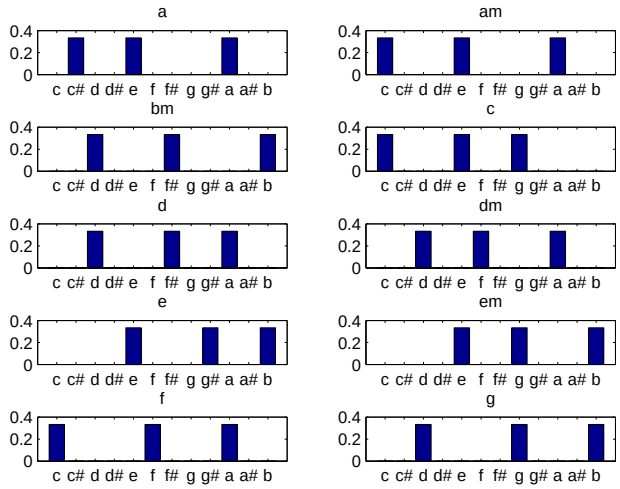


Figure 2. Ideal PCP representations of ten chords.

In order to compare PCP vectors, it is necessary to normalize them. Indeed, a chord can be played louder or softer and therefore, energy distribution can vary from one trial to another. To normalize a PCP vector, we divide the energy of each bin by the total energy of the original PCP, that is,

$$PCP(p) = \frac{PCP^*(p)}{\sum_{j=0}^{11} PCP^*(j)} \qquad (2)$$

where p is the index of the bin we want to normalize.

3.2. Experiments

The PCP is an intuitive descriptor for a musician because it highlights the main notes of a chord. Indeed, a musician is able to recognize a chord by identifying the notes contained in that chord. The PCPs of ten chords are given in Figure 2.

Apparently, the PCP seems to be suitable for representing a chord. However, recognizing chords based on the PCP is not a trivial task. In a first attempt, we developed a naive but simple chord detector that only compares histograms using a nearest neighbors (1-NN) method with the Bhattacharyya distance [4] as a distance measure. Basically, the algorithm takes an arbitrary PCP vector as input, normalizes it, and compares it to a predefined list of histograms representing the various chords to be recognized. The algorithm then classifies the chord as the one of the closest known histogram. It turns out that results of that simple method are unsatisfactory (see Section 5 for more details).

We also tried using variations of the PCP vector using 24-bin and 36-bin vectors. However, the overall results do not vary much. Therefore, we decided to keep the 12-bin vector for faster processing.

4. DATABASE

Part of the difficulties in machine learning techniques originate from the elaboration of a database of samples, also called dataset, and chord classification is no exception. The primary requirement for a chord recognizer using machine learning techniques is that the dataset contains enough data to build a model. Most of the work described in Section 2 on chord recognition does not make use of machine learning techniques which could explain why no chords database seems to be publicly available. For that reason, we had to create our own database of chords. In our database, we gathered audio files (recorded in the WAV format, sampled at $f_s = 44100$ Hz, and quantized with 16 bits), and the corresponding precomputed PCP vectors. The PCP vectors were computed on windows comprising each 16384 samples, which correspond to 0,37 seconds. The window size was chosen experimentally. We noticed that windows containing only 4096 samples produce correct results, however, best results for our application were achieved using a bigger window size.

Since our final goal is not to recognize all the existing chords, but to develop a music recognition system, we can limit chords to the most frequent ones. Therefore, we chose a subset of 10 chords:

$$A,\ Am,\ Bm,\ C,\ D,\ Dm,\ E,\ Em,\ F,\ G.$$

In our database, these chords are represented by an identifier ranging respectively from 0 to 9. Note that if other chords are also played in a song, the main chords can suffice. Moreover, many modern songs played in Western Europe are based on these 10 chords. Therefore, it seems to be a suitable starting point to validate our recognition method.

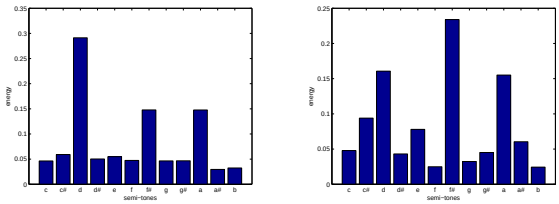
In practical terms, all PCP vectors are stored in a unique file which is organized as follows. Each line consists in a normalized PCP vector of twelve elements and one more element for the corresponding chord identifier. The following is illustrative of one line of the dataset file

$$0.04,\ 0.09,\ 0.18,\ 0.05,\ 0.12,\ 0.04,$$

$$0.14,\ 0.04,\ 0.03,\ 0.18,\ 0.04,\ 0.05,\ \mathbf{4}$$

The last digit corresponds to the class (the *D* chord in this example).

Next we concentrate on the context of the dataset. As we are willing to validate our dataset and test it on samples acquired in different contexts, the database was split into two subsets. The first subset contains a very large amount of guitar chord samples, whereas the second subset contains a smaller set of chords played with a different guitar and three other instruments. Therefore, there are two ways of using the database: we can either use cross-validation techniques on the first subset, or use it as a learning set while the second subset is used as a test set. Details follow.



(a) (b)

Figure 3. PCP representations of a D chord recorded with the same guitar in (a) an anechoic chamber, and (b) a noisy room. Note that the three major semi-tones are still visible in (b).

4.1. First subset

Chords of the dataset are produced with an acoustic guitar, which is probably the most common instrument in Western Europe to play chords. The acquisition conditions are the following. The chords samples were recorded in two different environments. Half of the chords were recorded in an anechoic chamber with a wideband microphone (01dB MCE320). The other half was recorded in a noisy environment, with a single live microphone (Shure SM58). We felt that samples recorded in both environments would reflect both the situations of professionals playing their songs in a studio and people playing music home. In Section 5, we derive that the system performs best if it is trained with a mix of noise-free and noisy chords. It is worth noticing that the chords were recorded using several playing styles (arpege, staccato, legato, etc.).

Figure 3 shows the PCP representations of a D chord recorded in the anechoic chamber and in the noisy room with the same guitar. As many real songs are played in a noisy environment, it is relevant to include noisy chords in the database.

For each chord, 100 samples were recorded in the anechoic chamber, and 100 samples were recorded in a noisy room. For each environment, the samples were recorded using four different guitars: one classical guitar with nylon strings, and three acoustic guitars producing three different sounds. It is expected that the variety of the dataset with respect to guitars will enhance the robustness of the system and extend its applicability, as there are many different guitar sounds available worldwide.

In conclusion, the first subset is organized as follows. There are 2000 chords in total. Each specific chord is recorded 200 times, 100 in an anechoic chamber and 100 in a noisy room. In both hundred halves, the chords are further separated into four subsets of 25 chords, produced with one of the four guitars.

4.2. Second subset

We also created a smaller database containing chords recorded with a guitar and three other instruments, namely a piano, a violin, and an accordion. That database is intended to provide an independent test set and should not be used to train the model. That database contains 100 chords for each instrument. These 100 chords are distributed equally among the ten chords mentioned earlier. Thus, there are 10 samples per chord for each instrument.

Parameters	Values
Number of hidden layers	1
Number of neurons in the hidden layer	35
Learning rate	0.001
Momentum	0.25
Weight decay	0.0

Table 1. Neural network parameters.

Our chords database is publicly available at *http://www.montefiore.ulg.ac.be/services/acous/STSI/file/jim2012Chords.zip*.

5. EXPERIMENTS

This section first introduces the learning method chosen for the design of our system. Then we detail the various experiments performed and discuss the results. With our experiments, we want to clarify and investigate several hypotheses:

1. We want to check if a naive application of the chord definition suffices.
2. How do we have to build the training set? Should noise-free samples, noisy samples, or both types of samples be included during training?
3. We want to evaluate the performance of our learning algorithm with our database.
4. Is the algorithm capable to recognize chords played with various other instruments?

5.1. Learning algorithm

Most techniques proposed in the literature for chord recognition do not use machine learning methods. Fujishima [9] used a pattern matching technique and heuristics to recognize chords. Although the techniques developed are efficient, they are complex. Since our final goal is not to develop a new chord descriptor, we chose to use a very simple, though powerful, technique to recognize chords. The chosen algorithm is a feed-forward neural network using a classical gradient descent algorithm with a negative log-likelihood [13] as cost function.

The neural network architecture is the following. There are twelve input attributes, which correspond to the twelve semi-tones of the PCP vector representing the chord. The neural network outputs a vector of 10 values, corresponding to the output neurons, each one being the probability of the detected chord to be issued from the corresponding chord. The final settings of the neural network were optimized by a 10-fold cross-validation on the learning database. Table 1 gives the parameters of the network.

TS / LS	Noise-free	Noisy	Mixed
Noise-free	4.0 %	5.0 %	4.0 %
Noisy	11.7 %	6.0 %	7.3 %

Table 2. Results of the validation with noise-free, noisy, and mixed Learning Sets (LS) and Test Sets (TS). The table gives the total classification error rate for each configuration.

5.2. Experiment 1: naive application of the chord definition

In this first experiment, we have created a synthetic and ideal sample of PCP for each chord manually (see Figure 2). Then, using the Bhattacharyya distance [4], we have applied a nearest neighbors (1-NN) algorithm on our second subset. The classification error rates obtained are the following: 8 % for guitar, 20 % for piano, 19 % for violin, and 32 % for accordion. These results are clearly unsatisfactory. The conclusion is straightforward: a learning based on real samples is necessary to reach the required performance level.

5.3. Experiment 2: determining the optimal learning set

In section 4.1, we explained that we created a database with noise-free chords and noisy chords. To justify that choice, we performed six tests to determine the best of the three following configurations:

- a learning set with noise-free chord samples only,
- a learning set with noisy chord samples only,
- and a learning set with mixed chord samples.

To perform the test, we split the database in different sets. First, the original database of 2000 samples was split into two sub-databases of 1000 elements. The first one only contains noise-free chords and the second one contains noisy chords. Then, we created three learning sets containing 70% of each sub-database. The first set contains 700 noise-free chords, the second 700 noisy chords, and the last one 350 noise-free chords and 350 noisy samples, taken randomly.

For the tests, we created two Test Sets (TS) with 30% of each sub-database. One test set contains noise-free chords and the second one contains noisy chords only. It is worth noticing that the chords used for the TS are not included in the LS.

Table 2 gives the results of each test. First, we trained the model with a noise-free learning set and tested it with the noise-free and noisy test sets (first column of the table). Next, we trained the model with a noisy learning set and tested it with a noise-free and noisy test sets (second column). Finally, we performed exactly the same tests with a learning set containing both noise-free and noisy chord samples (third column). The chords were recorded with different guitars distributed equally in each set.

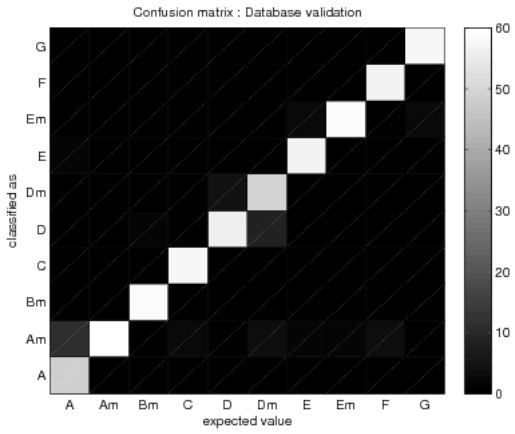


Figure 4. Confusion matrix for a network trained with a mixed learning set of 1400 samples (140 chords per class). The test set contains 600 samples (60 per class). The total error rate is 6.5%.

From the results given in Table 2, we conclude that building the model using a noise-free learning set produces the highest error rate for the noisy test set. Moreover, the optimal learning set (noisy or mixed) depends on the conditions under which the model has to be used. Unfortunately, we are not able to guess it in advance. However, we consider that the mixed learning set produces models that are less dependent of the noise in the database than with a noisy learning set, and therefore, we believe it is preferable to use a mixed learning set.

5.4. Experiment 3: validating the database

From our previous observations, we have decided to train our final model with both noise-free and noisy chord samples. Figure 4 shows the confusion matrix for the final network trained with a learning set of 1400 chords, that is 140 chord per class and a test set of 600 chords, that is 60 per class. The main database was split in two parts and thus, the result is slightly biased due to the size reduction of the database. Despite that bias, we can observe that the prediction of each class is quite good. Indeed, the rate of correct classification for each class is almost identical. Moreover, classification errors are not concentrated in a unique position.

5.5. Experiment 4: recognizing other instruments

In this experiment, we applied our method to other instruments. We chose four instruments capable of playing chords, namely a guitar, a piano, an accordion, and a violin. These instruments were chosen because they are widely used in Western Europe. Figure 5 compares the PCP representations of a C chord played with the four instruments. As can be seen, the PCP representations of the four instruments are similar.

Although the model was trained with a database containing only guitar chords, we applied it on the indepen-

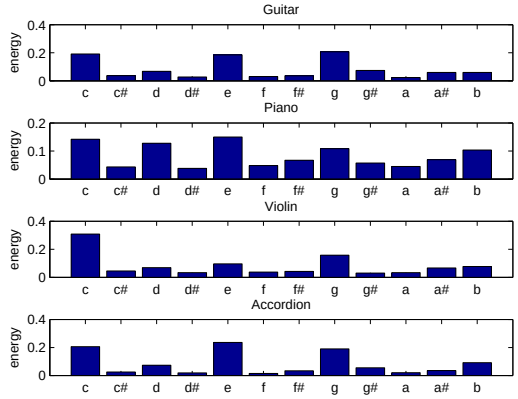


Figure 5. PCP of a C chord played with four instruments (respectively, from top to bottom, a guitar, a piano, a violin, and an accordion). The c, e and g notes are very similar on the four instruments and dominate over the other notes.

Instrument	1-NN with the chord definition	Neural network with a learning set
Guitar	8 %	1 %
Piano	20 %	13 %
Violin	19 %	5 %
Accordion	32 %	4 %

Table 3. Classification error rates for 4 different instruments using our method.

dent test sets mentioned in Section 4. It is worth remembering that these recordings are completely independent of the chords used to train the model.

Table 3 gives the results of our method for the four instruments, and compares them to the naive approach (see Section 5.2). We observe huge improvements in the results with a learning method based of real chord samples compared to that of the naive approach. It appears that it is harder to recognize chords played on a piano, which could be explained by the noisy nature of piano sounds (as graphically illustrated with the PCP of a piano, in Figure 5).

Figure 6 shows the confusion matrices obtained for each instrument using the trained neural network. The predictions are good, but less precise for the piano. Best results are obtained with an independent test set of guitar chords, as the model was trained with guitar chords. Violin and accordion also give good results compared to the naive method, and produce a classification error rate of respectively 5% and 4%, which is promising.

6. CONCLUSIONS

This paper presents a method based on machine learning techniques, using a feed-forward neural network, for chords recognition, applicable to raw audio. As no chords database seems to be publicly available, we have built our own dataset containing chords recorded in an anechoic chamber and in a noisy room. Both environments were chosen to increase the robustness of the algorithm with respect to the acquisition process. The database is made of 2000 different guitar chords saved in WAV files and distributed into 10 chords classes.

We have highlighted that the best strategy consists in using a learning set containing both noise-free and noisy samples. Experimental results also show that our attributes, that is, the 12-dimensional PCP vectors, are effective representations of chords, and that they are also applicable to other instruments, like piano, violin, and accordion. However, the PCP representation has to be sent to a feed-forward neural network which learns a model to recognize the ten chords. Our method, based on the provided database, outperforms a direct application of the chord definition using 1-NN with Bhattacharyya distance. Finally, results for the recognition of chords played with other instruments are also presented. Our method also performs well for such PCP samples.

Despite the use of a simple 12-bin PCP vector based on the Discrete Fourier Transform, we show promising results and fast processing, which would have probably not been achieved with more complex pre-processing steps. It is worth noticing however, that for pure chords identification, our system is limited to ten chords, which may seem restrictive. Thus, the lack of availability of a complete labeled chords database is a limitation to our system.

However, recognizing ten chords seems to be sufficient for our next application, i.e., song excerpts recognition. Moreover, fast processing is an important property of this application and the first results are very promising.

In this future work, we also plan to consider other algorithms for chords recognition, particulary in relation with the recently released Million Song Dataset (MSD) [2] to improve chords and music recognition.

Acknowledgments

We would like to thank Maroussia OSMALSKYJ, Maxime MICHALUK and Philippe ELOY for helping us recording the piano, the violin and the accordion.

7. REFERENCES

[1] J.-J. Aucouturier and F. Pachet. Music similarity measures: What's the use? In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, pages 157–163, 2002.

[2] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[3] M. Carre. *Systèmes de Recherche de Documents Musicaux par Chantonnement*. PhD thesis, École Na-

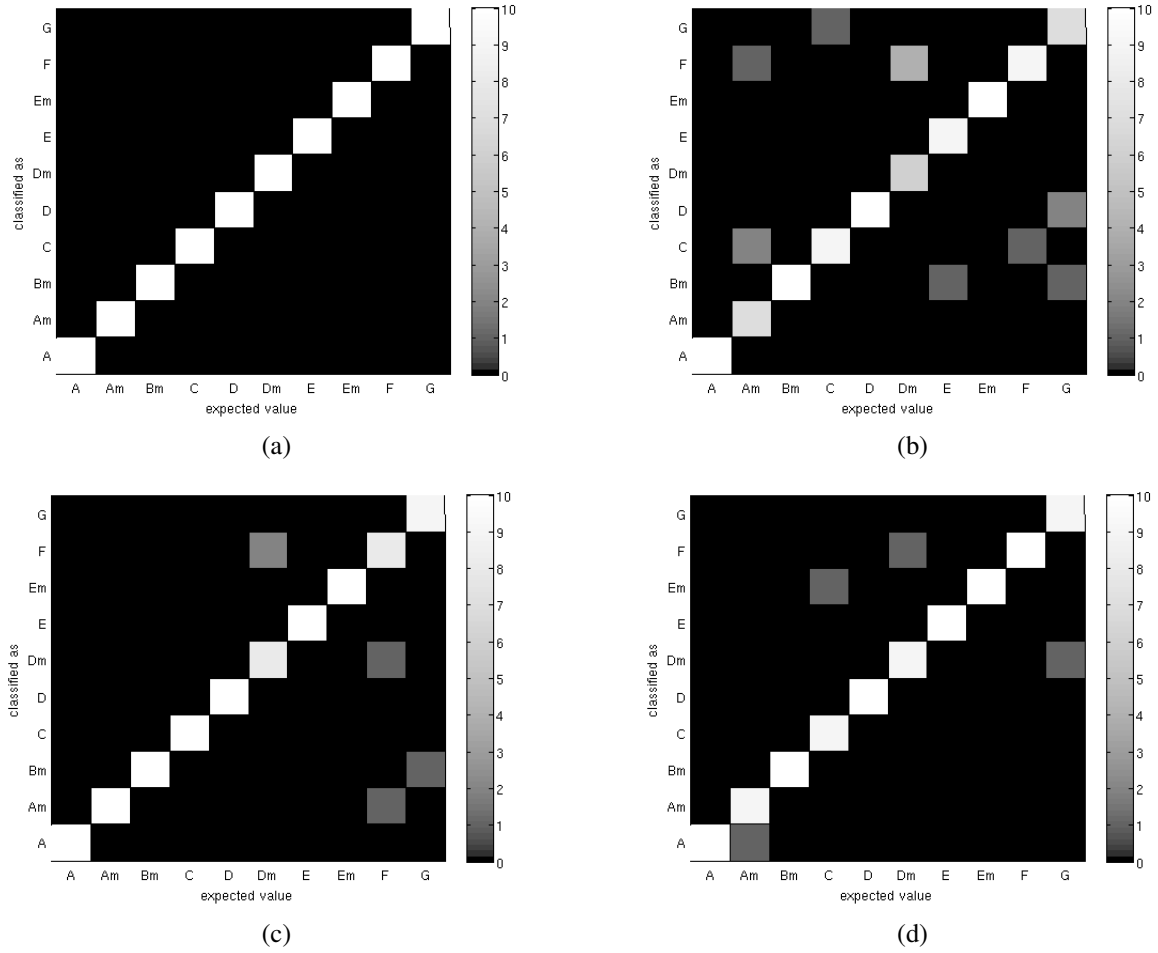


Figure 6. Confusion matrices of tests performed for four instruments: (a) guitar, (b) piano, (c) violin, (d) accordion.

tionale Supérieure des Télécommunications, Paris, 2002.

[4] M. Deza and E. Deza. *Encyclopedia of Distances*. Springer, 2009.

[5] A. Egorov and G. Linetsky. Cover song identification with if-f0 pitch class profiles. In *MIREX extended Abstract*, 2008.

[6] D. Ellis and C. Cotton. The 2007 labrosa cover song detection system. In *MIREX Extended Abstract*, 2007.

[7] D. Ellis and G. Poliner. Identifying cover songs with chroma features and dynamic programming beat tracking. In *Proceedings of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2007.

[8] A. Flexer, D. Schnitzer, M. Gasser, and G. Widmer. Playlist generation using start and end songs. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 173–178, 2008.

[9] T. Fujishima. Realtime chord recognition of musical sound: a system using common lisp music. In *Proceedings ot the International Computer Music Conference (ICMC)*, pages 464–467, 1999.

[10] E. Gomez and P. Herrera. Automatic extraction of tonal metadata from polyphonic audio recordings. In *Proceedings of the International Conference: Metadata for Audio*, 2004.

[11] E. Gomez and P. Herrera. The song remains the same; identifying versions of the same song using tonal descriptors. *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 180–185, 2006.

[12] C. Harte and M. Sandler. Automatic chord identifcation using a quantised chromagram. In *Proceedings of the 118th Audio Engineering Society Convention (AES)*, 2005.

[13] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Series in Statistics. Springer, second edition, September 2009.

[14] J. Jensen, M. Christensen, D. Ellis, and S. Jensen. A tempo-insensitive distance measure for cover song identification based on chroma features. In *Proceedings of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2209–2212, 2008.

[15] S. Kim and S. Narayanan. Dynamic chroma feature vectors with applications to cover song identification. In *Proceedings of the IEEE Workshop on Multimedia Signal Processing (MMSP)*, pages 984–987, October 2008.

[16] K. Lee. Automatic chord recognition using enhanced pitch class profile. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 306–313, 2006.

[17] M. Müller, F. Kurth, and M. Clausen. Audio matching via chroma-based statistical features. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 288–295, 2005.

[18] J. Serra, E. Gómez, and P. Herrera. Audio cover song identification and similarity: Background, approaches, evaluation, and beyond. In *Advances in Music Information Retrieval*, pages 307–332. Springer, 2010.

[19] A. Sheh and D. Ellis. Chord segmentation and recognition using EM-trained hidden markov models. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 121–124, 2003.

[20] L. Xiao, L. Liu, F. Seide, and J. Zhou. Learning a music similarity measure on automatic annotations with application to playlist generation. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 1885–1888, 2009.

DETECTION DES TECHNIQUES DE JEU DE LA GUITARE

Otso Lähdeoja
Institut numediart,
TCTS Lab, UMONS
Mons, Belgique
otso.lahdeoja@free.fr

Loïc Reboursière
Institut numediart,
TCTS Lab, UMONS
Mons, Belgique
loicreboursiere@gmail.com

Thomas Drugman
Institut numediart,
TCTS Lab, UMONS
Mons, Belgique
thomas.drugman@umons.ac.be

Stéphane Dupont
Institut numediart,
TCTS Lab, UMONS
Mons, Belgique
stephane.dupont@umons.ac.be

Cécile Picard-Limpens
Institut numediart,
TCTS Lab, UMONS
Mons, Belgique
ccl.picard@gmail.com

Nicolas Riche
Institut numediart,
TCTS Lab, UMONS
Mons, Belgique
nicolas.riche@umons.ac.be

RÉSUMÉ

Dans cet article nous présentons un ensemble d'approches pour une détection de sept techniques de jeu de la guitare: légatos ascendants et descendants, notes tirées, glissées, étouffées ainsi que la position d'attaque sur la corde. Ces algorithmes ont été conçus pour permettre au guitariste un contrôle direct des données audionumériques via ses techniques de jeu habituelles. Le but du projet est de fournir un ensemble intégré et dédié à la détection des techniques de jeu, adaptable aux différentes guitares et styles de jeu. Nous présentons ici des résultats intermédiaires mettant en évidence les algorithmes développés et leurs performances, ainsi que les premiers essais d'implémentation en une application temps réel.

1. INTRODUCTION

La connexion entre les instruments issus de la tradition acoustique et les systèmes informatiques est un sujet porteur de perspectives fertiles pour la création musicale. L'instrument de musique peut être, dorénavant, vu comme une interface permettant un contrôle extrêmement fin d'évènements sonores. L'accès à l'intimité des gestes du musicien offre des données en lien direct avec le discours musical ainsi qu'avec la corporéité de l'instrumentiste propres à être utilisées afin d'augmenter la performance avec des dispositifs audionumériques. La captation des gestes de l'instrumentiste peut être effectuée de deux manières: 1) directement par des capteurs disposés sur le corps du musicien ou sur l'instrument, ou 2) par l'analyse du signal audio de l'instrument, résultant des gestes du musicien. On dénomme ces techniques respectivement captation directe et indirecte [15].

A l'Institut numediart de l'Université de Mons, nous avons développé un projet intitulé "Guitar as Controller", afin d'explorer les possibilités d'une guitare augmentée par un système de contrôle gestuel de données numériques. Ce projet a été dédié à l'étude de la captation indirecte sur la guitare et a eu pour but, comme son nom l'indique, de transformer l'instrument en contrôleur en développant des algorithmes de détection pour sept des principales techniques de jeu de la guitare.

2. TRAVAUX DE REFERENCE

La captation des techniques de jeu d'instrument acoustique a suscité un grand nombre de travaux, souvent portés sur les instruments à cordes frottées, mais aussi sur la guitare. Les techniques de l'archet du violon ont été étudiées en détail, par l'équipe de Tod Machover au MIT [6] et par l'équipe du violon augmenté de l'IRCAM [12].

Concernant la guitare, le "Guitar Lab" de l'Université de Malaga a étudié certaines techniques de jeu de la main gauche, aboutissant à des algorithmes de détection fiables en temps différé [9]. Caroline Traube [14] a effectué des travaux sur la détection de la position d'attaque, complétés par les travaux de l'équipe du TKK en Finlande [10]. Paul O'Graddy [8] a investigué la transcription automatique de la guitare à partir d'un signal hexaphonique (1 signal audio par corde). D'autres travaux se focalisent plus sur le côté artistique de la guitare augmentée soit par des techniques de captation directe ou indirecte utilisées pour contrôleur des paramètres de synthèse sonore: [5] [11] [13]. Après avoir présenté sa guitare hexaphonique augmentée, Adrian Freed [3] va jusqu'à introduire la guitare sans corde.

Le but de notre projet a été d'inclure toutes les techniques "majeures" du jeu de la guitare au sein de nos recherches, et de parvenir à des algorithmes pouvant fonctionner dans le contexte du "temps réel musical", c'est-à-dire avec une latence inférieure à 30 ms. Nous souhaitons ainsi contribuer à l'établissement d'une technologie de guitare augmentée similaire au principe de la guitare MIDI, en proposant une plus grande finesse dans les articulations guitaristiques détectées.

3. METHODOLOGIE DE TRAVAIL

Nos travaux ont été effectués sur une guitare Godin Multiac, équipée de cordes nylon/alliance Savarez Alliance HT classic et d'un capteur RMC hexaphonique piézoélectrique positionné juste au niveau du sillet. Nos algorithmes de détection opèrent de manière individuelle sur les six cordes de l'instrument. L'hexaphonie permet un suivi individuel de chaque corde lors du jeu polyphonique, permettant ainsi une finesse d'analyse beaucoup plus élevée qu'une approche monophonique. La méthodologie du travail a fait se succéder les phases de 1) choix des techniques de jeu, 2) création d'une base de données, 3) développement d'un algorithme de détection d'événements (*onset*) et de discrétisation d'attaques main droite et main gauche, 4) détermination des algorithmes pour chaque technique de jeu.

3.1. Techniques de jeu

Nous avons choisi un ensemble de sept techniques de jeu comme base de travail. Ces techniques correspondent aux modes de jeu les plus courants de la guitare moderne et excluant certaines techniques jugées virtuoses. Le choix des techniques de jeu s'est appuyé sur le référencement de Jonathan Norton [7].

Légato ascendant (*hammer-on*): note martelée avec un doigt de la main gauche
Légato descendant (*pull-off*): note tirée avec un doigt de la main gauche
Glissé (*slide*): note attaquée avec la main droite, dont la hauteur est altérée par la main gauche en glissant sur le manche
Tiré (*bend* ?): note attaquée avec la main droite, dont la hauteur est altérée par la main gauche en tirant sur la corde
Notes harmoniques: note pincée avec la main droite, légèrement effleurée au-dessus d'un nœud harmonique par la main gauche
Position d'attaque sur la corde (*plucking point*): localisation attaques main droite entre le chevalet et le manche (*ponticello - sul tasto*)
Notes étouffées avec la paume (*palm mute*): note attaquée par la main gauche tout en étouffant avec la paume, produisant un effet "sourdine".

3.2. Base de données

Le développement de nos algorithmes a été effectué à partir d'une base de données exhaustive des techniques de jeu de la guitare, créée pour cette occasion. Toutes les techniques mentionnées ci-dessus ont été enregistrées par deux guitaristes avec deux techniques de jeu différentes: l'un jouant aux doigts, l'autre au médiator. La base de données a été constituée en suivant les points listés ci-dessous :

• Les notes, de la corde à vide jusqu'à la 16ème frette ont été enregistrées
• Les légatos ont été enregistrés sur tout le manche en variations d'un demi-ton, d'un ton, et d'une tierce mineure.
• Les glissés ont été enregistrés en ascendant et en descendant
• Les notes tirées ne sont jouées qu'au demi-ton, les cordes en nylon ne permettant pas davantage d'amplitude
• Les cinq premières harmoniques ont été enregistrées pour chaque corde
• Les attaques "normales" ont été enregistrées en trois positions: chevalet, rosace et manche.

Chacun des guitaristes a enregistré 1416 éléments de jeu (288 notes normales, 568 légatos, 468 glissés, 66 tirés, 30 harmoniques).
La base de données que nous avons enregistrée est disponible en ligne[1].

4. FONCTIONNEMENT DU SYSTEME DE DETECTION

La guitare est un instrument à cordes pincées, toute production du son y est liée à un évènement d'excitation de la corde. Notre système prend comme unité de départ la mise en vibration de la corde (*onset*). A partir de la détection d'un nouvel évènement, une discrimination est opérée afin de déterminer si cet évènement provient de la main droite ou de la main gauche (type d'attaque). A partir de cette information nous définissons le type d'articulation employé. Pour le moment, notre système n'est pas conçu pour la détection d'une note conjuguant deux types d'articulation simultanés (par exemple une note étouffée à la main droite et glissée à la main gauche).

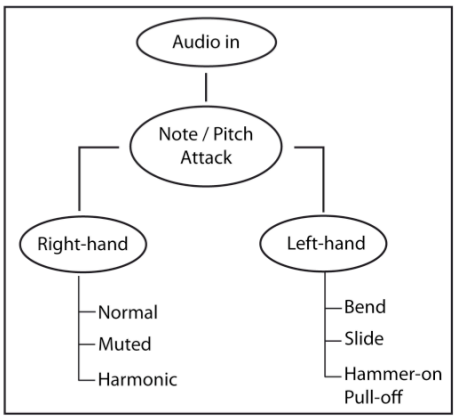


Figure 1. Schéma général du système de détection de techniques de jeu.

4.1. Détection d'évènement (*onset*)

Le canevas général de la détection d'un évènement (*onset*) s'articule autour de 3 étapes [1]. Tout d'abord le signal audio est pré-traité pour accentuer certains éléments-clés pour la détection (filtrage, séparation en

[1] http://www.numediart.org/GuitarDB/

bandes de fréquence, séparation entre les transitoires d'attaque et les portions stables du signal). Dans une seconde étape, la quantité d'information est réduite pour obtenir un débit de traitement plus faible. Pour finir, une étape de seuillage ou de détection de pics est utilisée pour isoler les *onsets* potentiels. Plus d'informations sur la détection d'onset peut-être trouvé dans [2].

Dans le cadre de ce projet, nous avons comparé plusieurs approches de détection d'onsets sur notre base de données de la guitare. L'évaluation suit la méthode préconisée dans [4] avec une tolérance de plus ou moins 50ms. Une tolérance aussi élevée est nécessaire pour palier au timing approximatif de la segmentation effectuée manuellement. Cependant, pratiquement parlant, les méthodes que nous comparons ici restent plus précises que la segmentation manuelle, l'erreur de détection restant généralement en dessous d'une demi fenêtre d'analyse (10 ms dans notre cas).

Les méthodes que nous avons comparées couvrent 18 variantes de méthodes basées soit sur l'amplitude soit sur la transformée de Fourier à court terme (incluant des techniques de flux spectral, des techniques basées sur la phase et leurs variantes utilisant des amplitudes pondérées ainsi que des techniques dans le domaine complexe). Il faut noter que nous n'avons pas utilisé de méthodes nécessitant une estimation de hauteur de note (*pitch*). Les fichiers de guitare utilisés pour ce test étaient les enregistrements monophoniques (somme des 6 signaux séparés) des techniques de jeu suivantes : legato ascendant, legato descendant, harmonique ainsi que les notes jouées normalement (attaque main droite). Des seuils de détection pour la F-mesure et pour le rappel (*recall*) ont été optimisés pour chaque méthode individuellement.

Nos résultats montrent que si pour les notes produites avec la main droite, la détection n'est, et ceci peu importe le type de détection employée, pas un problème, la détection des legato ascendants et descendants est meilleure lorsqu'une approche de flux spectral basée sur l'amplitude des points (bins) de la transformée de Fourier à court terme est utilisée. Elle produit une F-mesure supérieure à 96% avec un rappel proche de 100% pour les attaques main droite et pour les harmoniques, de 99% pour les légatos ascendants. Les légatos descendants montrent un résultat légèrement moins bon de 88%. Plus d'efforts devront portés sur ce dernier type de technique pour comprendre comment obtenir une meilleure détection.

4.2. Articulation entre la main droite et la main gauche

Lorsqu'une corde est pinçée, le doigt ou le médiateur arrête son régime vibratoire pour un court moment avant sa mise en vibration. Au niveau de la forme d'onde du signal, cela se traduit par une courte mais marquée baisse dans l'amplitude suivie de l'enveloppe d'attaque (Figure 2).
A l'inverse, les attaques légato de la main gauche n'arrêtent pas la vibration de la corde, produisant un changement de la hauteur de note sans baisse d'amplitude significative avant l'apparition de la nouvelle note (Figure 2).
Notre système de différenciation main droite/main gauche est donc basé sur la détection de cette baisse significative de l'amplitude en analysant les quelques millisecondes précédant un évènement sonore (*onset*).

Une mesure simple de la pente entre le point d'énergie minimum précédant l'attaque et le point maximum au moment des transitoires d'attaque est effectuée. Avec un seuil optimisé, nous arrivons à un taux de réussite de 94% sur notre base de données. Les erreurs de classification sont la plupart du temps dues à des bruits parasites accompagnant l'attaque. Une autre limitation de notre système apparaît lors d'un jeu très rapide, où la vitesse des attaques brouille la fiabilité des mesures entre les points minimum et maximum de la forme d'onde.

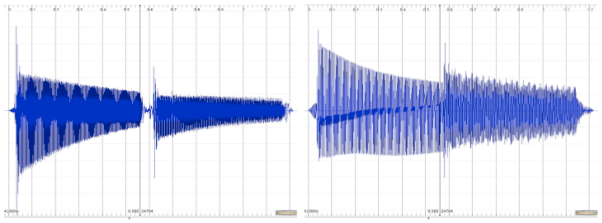


Figure 2. A gauche, deux notes consécutives attaquées à la main droite. On remarque une baisse de l'amplitude avant la nouvelle attaque. A droite, la deuxième note est jouée légato avec la main gauche. La vibration de la corde ne subit pas d'arrêt momentané.

5. TECHNIQUES DE JEU DE LA MAIN GAUCHE

Une fois l'évènement classé dans la catégorie "main gauche", il convient de déterminer la technique de jeu employée. Pour ce faire, nous analysons l'évolution temporelle du profil de hauteur (fréquence) de la note dans la durée. En effet, les articulations de la main gauche opèrent sur le placement des doigts sur les frettes ou sur la tension de la corde. Elles modifient le paramètre de hauteur dans le temps, chacune ayant un profil temporel spécifique.

Notre système mesure l'évolution de la hauteur en fonction du temps et du nombre de demi-tons de la transition entre deux notes. Après la détection de l'attaque initiale, la fréquence est mesurée par des fenêtres de 10 millisecondes et le résultat divisé par la fréquence initiale. Le système cherche la pente maximale entre deux fenêtres successives. La figure 3 montre la répartition des pentes maximales par demi-tons pour les

techniques de jeu de légato ascendant et descendant, le tiré et le glissé.

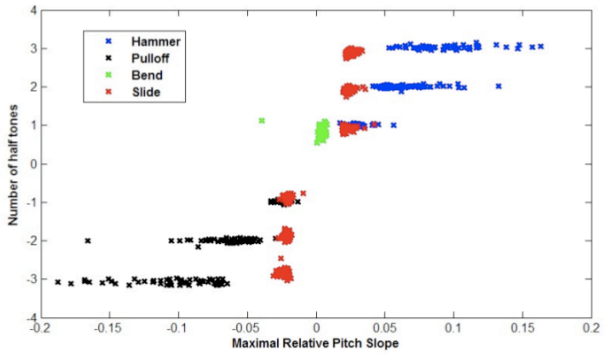


Figure 3. Distribution des légatos ascendants (bleu), descendants (noir), tirés (vert), et glissés (rouge) selon la dérivée de la hauteur et le nombre de demi-tons.

5.1. Légato ascendant et descendant (*hammer-on* et *pull off*)

Le jeu en légato consiste à modifier la hauteur par un martelé (ascendant, bleu dans la Figure 3) ou pincé (descendant, noir dans la Figure 3) de la main gauche. Dans les deux cas, le profil de fréquence est marqué par une pente abrupte qui se démarque clairement pour les intervalles supérieurs au demi-ton.

5.2. Tiré (*bend*)

Une note tirée (vert dans la Figure 3) affiche une pente relativement lente et obligatoirement ascendante formant un groupe identifiable sur la Figure 3. Il faut noter qu'à cause du type de cordes (nylon-alliage) utilisé, nous n'avons inclus que des notés tirées au demi-ton dans notre étude.

5.3. Glissé (*slide*)

Un *glissando* sur la guitare produit un profil similaire à la note tirée, mais avec une évolution un peu plus rapide (rouge dans la Figure 3). La pente des glissandi se situe entre celle des légato et des tirés, clairement identifiable sur la Figure 3 pour les intervalles supérieurs au demi-ton. Un marqueur supplémentaire peut être constitué par le fait que l'évolution de la hauteur des glissandi affiche des discontinuités correspondant aux frettes, là où le tiré affiche une continuité dans son profil. Toutefois nous n'avons pas implémenté la recherche de ce marqueur dans notre système.

Par la comparaison des pentes hauteur/temps, nous arrivons ainsi à classer toutes les techniques de la main gauche pour des intervalles supérieurs au demi-ton. Toutefois, le cas des demi-tons est problématique, du au fait que les vitesses d'exécution d'un glissé, d'un tiré et d'un légato d'un demi-ton sont similaires. Pour pallier à ce problème, nous avons testé un procédé supplémentaire afin de distinguer les notes glissées par leur fort contenu en hautes fréquences (du aux bruits de frottement du doigt passant sur les frettes). Pour ce faire, nous comparons le rapport entre le centre de gravité spectrale et le taux d'énergie de l'attaque. En combinant ces deux approches, nous arrivons à une classification satisfaisante, avec un taux de détection entre 93% et 100% selon la technique étudiée.

	Hammer-on	Pull-off	Bend	Slide
Hammer-on	93.27%	0.45%	0%	6.28%
Pull-off	0%	93.69%	0%	6.31%
Bend	0%	0%	100%	0%
Slide	1.74%	0.43%	0.21%	97.61%

Table 1. Matrice de confusion de la détection des quatre techniques de la main gauche (légatos ascendant et descendant, tiré et glissé).

6. TECHNIQUES DE JEU DE LA MAIN DROITE

6.1. Notes étouffées avec la paume

Le *palm muting* fait intervenir la paume de la main droite pour produire un effet étouffé, similaire au *con sordino* des cordes frottées. Les caractéristiques des notes étouffées comparées aux notes normales sont : une durée plus courte, une baisse plus rapide d'énergie globale dans l'enveloppe, ainsi qu'un faible taux d'harmoniques aigues. En tenant compte de ces marqueurs, nous avons développé un algorithme qui calcule la baisse d'énergie dans le spectre au dessus de 500Hz à partir de l'attaque. La partie grave/bas-médium du spectre (<500Hz) est filtrée pour clarifier l'analyse car cette zone contient beaucoup d'énergie pour toutes les techniques de jeu et la baisse d'énergie caractéristique du *palm muting* y est, de fait, moins visible. Les résultats sont regroupés par

String	Normal notes	Palm muted notes
1 (*thresh -0.06*)	97.91%	93.75%
2 (*thresh -0.06*)	100%	100%
3 (*thresh -0.06*)	100%	100%
4 (*thresh -0.04*)	100%	100%
5 (*thresh -0.05*)	100%	100%
6 (*thresh -0.05*)	97.91%	87.5%

corde dans le tableau 2. Notre système affiche un taux moyen de détection de 99,3 pour les notes normales et 96,9 pour les notes étouffées.

Table 2. Taux de détection des notes étouffées (par corde).

6.2. Estimation de la position à laquelle la corde à été pincée

Lorsqu'une corde est pincée, deux faits physiques caractéristiques apparaissent :

- Le spectre de la note montre une déficience en harmoniques qui ont un nœud (*node*) à l'endroit ou la corde a été pincée
- Deux ondes parcourent la corde en sens opposé

Les deux méthodes [14] [10] présentes dans la littérature sur cette question utilisent respectivement ces deux points.

L'approche que nous proposons se base sur le 2ème fait cité ci-dessus. Ce dernier a une conséquence sur le comportement du signal dans les quelques millisecondes suivants l'attaque. La figure 4 montre les formes d'ondes spécifiques pour les trois positions : chevalet (~2cm), rosace (~10cm), manche (XIXème frette, ~22cm). Nous pouvons noter que la position du minimum de la fenêtre (1.5 période à partir de l'attaque) présentée sur cette figure évolue en fonction de l'endroit où la corde est pincée. Notre algorithme est basé sur le calcul de la distance entre ce minimum et la position de la période du signal par rapport à l'attaque.

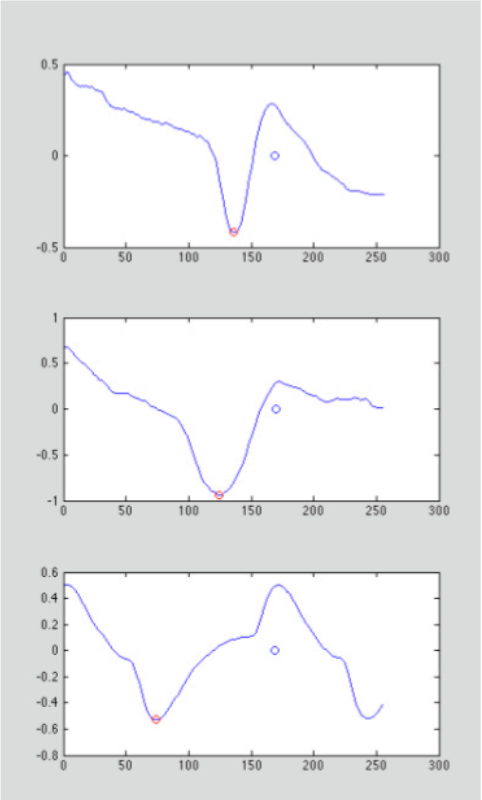


Figure 4. Formes d'onde d'une période et demie de trois attaques: chevalet, rosace et manche. En rouge le point minimum et en bleu la position de la période sur la forme d'onde.

La figure 5 montre l'histogramme résultant du calcul de cette distance pour les fichiers joués au chevalet, à la rosace et au manche. La distance est normalisée. Une valeur de 0 correspond à la position au chevalet alors qu'une valeur de 0.5 correspond à la position de la XIIe frette (moitié de la corde à vide).

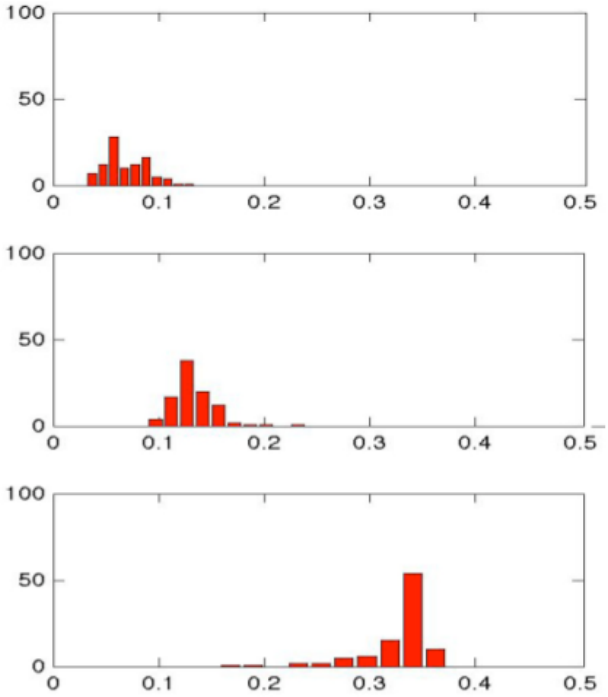


Figure 5. Histogramme des détections d'attaque conduites sur notre base de données. En abscisse la distance normalisée, en ordonnées la quantité de détections par position de jeu.

Exception faite de quelques erreurs, la position manche semble clairement se démarquer des deux autres. Les positions chevalet et rosace présentent à priori, un recouvrement plus important. Cependant, en fixant précisément deux seuils délimitant chacune des trois zones, la précédente conclusion est nuancée. Le tableau 3 présente les résultats obtenus après applications de ces deux seuils. Tous fichiers confondus, un maximum de 8% d'erreur est constaté. Ces erreurs peuvent être en partie dues au fait que la position du pincement n'a pas été précisément mesurée lors des enregistrements.

	$d < 0.1$	$0.1 < d < 0.233$	$d > 0.233$
Bridge	98%	2%	0%
Soundhole	4%	96%	0%
Fretboard	0%	2%	98%

Table 3. Résultats de la détection de position d'attaque sur la corde sur 3 points: chevalet, rosace et manche.

6.3. Notes Harmoniques

Les notes harmoniques sont produites en effleurant la corde avec la main gauche à un nœud harmonique pendant une attaque main droite. A l'oreille, le timbre d'une note harmonique est aisément reconnaissable. Toutefois, la reconnaissance des harmoniques par un algorithme ne va pas de soi. Les modèles de détection que nous avons testé ont été confrontés au problème de distinction entre les notes harmoniques et les attaques sur

le manche (*sul tasto*). En effet, les profils spectraux de ces deux modes de jeu sont fort similaires.

La technique que nous avons retenue est basée sur le taux harmonique - subharmonique (*Harmonic-to-Subharmonic ratio*), calculé sur une fenêtre de 40 ms après l'attaque. Le taux calcule la différence d'énergie en dB entre la première harmonique (*F*0) et la subharmonique (1,5 x *F*0). La Figure 6 montre des profils spectraux typiques pour une note normale et une note harmonique, ainsi que les points correspondants à l'harmonique et au subharmonique. La valeur du taux harmonique - subharmonique est généralement plus bas pour les notes harmoniques.

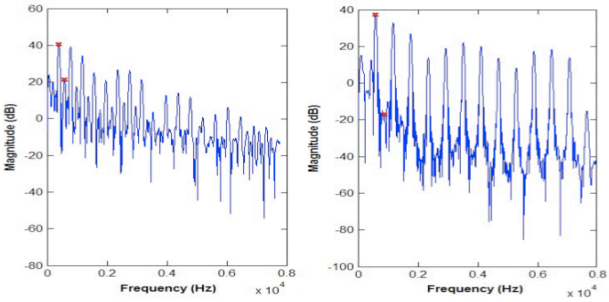


Figure 6. Spectre d'une note harmonique (gauche) et normale (droite) au moment de l'attaque.

Nos résultats montrent une distinction nette entre les notes harmoniques et normales, sauf pour les attaques effectués en position de manche. D'une manière générale, ce sont surtout les deux premières harmoniques (frettes 7 et 12) qui posent problème. Pour améliorer cette détection nous avons testé un système fonctionnant dans le domaine temporel. Il s'agit de la mesurer deux données de l'attaque: la durée de l'attaque, définie comme le temps que la forme d'onde reste positive après l'attaque, ainsi que la discontinuité relative, definie comme la différence entre les points Amin et Amax de l'attaque. La figure 7 met en évidence ces deux données. Nous constatons que les notes harmoniques ont généralement une plus grande durée d'attaque et une plus important discontinuité relative.

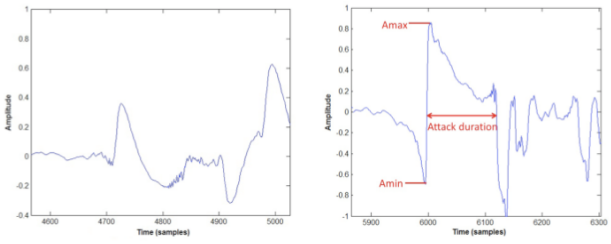


Figure 7. Enveloppe d'attaque d'une note normale (gauche) et harmonique (droite). Les point Amin et Amax servant à calculer la discontinuité relative ainsi que la durée d'attaque sont également indiqués.

En combinant ces deux approches, nous arrivons à des taux de détection au-dessus de 94%, détaillés dan le tableau 4.

	Harmonic detection	Normal detection
Harmonic	100%	0%
Fretboard	0.52%	99.48%
Soundhole	2.78%	97.22%
Bridge	5.38%	94.62%

Table 4. Matrice de confusion de la détection des notes harmoniques et les attaques chevalet, rosace, manche.

7. MAPPING

Notre projet de détection des techniques de jeu de la guitare a été motivé par la volonté d'augmenter la guitare par un contrôle gestuel des média numériques. Pour une première illustration des possibilités du système, nous avons implémenté des mises en correspondance (*mapping*) entre certaines techniques et des effets sonores. Nous nous rapprochons ainsi de l'idée d'un guitariste contrôlant ses effets par des gestes directement liés au jeu de l'instrument. Les mappings que nous avons effectués sont:

• Notes tirées : effet de filtre wah-wah: le guitariste contrôle le fréquence du filtre par la hauteur de la note tirée.

• Notes étouffées : effet "octaveur" rajoutant une octave grave au son. Cet effet permet au guitariste d'accéder aux sonorités d'une guitare basse par simple étouffement des cordes.

• Légato ascendant et descendant : effet percussif par la convolution du son de la guitare avec un son préenregistré (un *slap* de clarinette dans notre cas).

Nous avons également réalisé un mapping visuel vers une représentation en 3D du manche, mettant en évidence les techniques de jeu par un feedback visuel. Une vidéo de démonstration de notre système est visible à : http://www.numediart.org/demos/ (guitar as a controller)

8. ETAT DES TRAVAUX, PROBLEMATIQUES, PERSPECTIVES

Le projet de 6 mois mené à l'institut numediart a permis de développer un algorithme fonctionnel et prometteur pour chacune des sept techniques de jeu sélectionnées. L'ensemble de ces algorithmes fonctionnent en temps différé sur Matlab, mais ils sont techniquement portables en temps réel. En outre, nous avons développé une série de patchs Max MSP pour tester nos concepts dans des conditions du live. Les techniques de légato, tiré, étouffé, et point d'attaque ont été portés et testés sur Max MSP, et fournissent une preuve de concept pour notre système. Toutefois, des problèmes de calcul apparaissent lors qu'on utilise l'ensemble de nos algorithmes en simultané. Pour avancer vers un système intégré, il nous semble

nécessaire de passer sur une autre plateforme de programmation offrant plus de contrôle sur les opérations de bas niveau. Au-delà de l'aspect logiciel, nous avons exploré la possibilité de porter notre système sur un hardware dédié, équipé d'une entrée/conversion A/D hexaphonique et d'un FPGA. Sur le moyen terme, nous souhaiterions concevoir un système hexaphonique "*plug and play*" qui puisse servir à la fois d'interface audio pour la guitare ainsi que d'analyseur de techniques de jeu. Les informations provenant de l'analyse seraient envoyées à l'ordinateur (ou autre dispositif audionumérique) sous forme de données OSC ou MIDI. Ces travaux sont en cours et nécessitent encore d'amples recherches avant d'être pleinement opérationnels.

9. CONCLUSION

Dans cet article, nous avons présenté un ensemble d'algorithmes de détection de techniques de jeu de la guitare par l'analyse du signal hexaphonique. Des solutions opérationnelles ont été trouvées pour sept techniques, couvrant la totalité des modes de jeu les plus utilisés de l'instrument. Les taux de détection se situent systématiquement autour de 95% de réponse correcte. Pour le moment, une partie importante de nos algorithmes ont été portés et testés sur max/MSP dans les conditions du temps réel musical, et ils montrent des résultats directement applicables pour la création musicale. Afin de poursuivre nos recherches vers une guitare augmentée permettant de contrôler des données audionumériques via les techniques de jeu, nous souhaitons désormais concentrer nos efforts vers la mise en place d'un système logiciel dédié, fonctionnant en temps réel et couvrant l'ensemble des techniques étudiées.

10. REFERENCES

[1] Bello J. P., Daudet L. *et al*. A tutorial on onset detection in music signals. IEEE Transactions on Speech and Audio Processing,, 16 (5), 2005.

[2] Brossier P.M. Automatic Annotation of Musical Audio for Interactive Applications, Phd thesis, Centre dor Digital Music Queen Mary, University of London, august 2006.

[3] Freed A. Joy of hex and the cordless guitars of the future. In Identites de la Guitare Electrique, 2009.

[4] Holzapfel A Three dimensions of pitched instrument onset détection,.Holzapfel, IEEE Transactions on Audio, Speech and Language Processing, 18 (6), 2010.

[5] Lähdeoja O. Une approche de l'instrument augmentée, La guitare éectrique. PhD thesis, Ecole Doctorale Esthétique, Sciences et Technologies des Arts, Université Paris 8, 2010.

[6] Machover T. *Hyperinstruments – A Progress Report 1987-1991*, Technical report, Massachusetts Institute of Technology, 1992.

[7] Norton J. C. *Motion capture to build a foundation for a computer-controlled instrument by study of classical guitar performance*, Thèse de doctorat, CCRMA, Dept. of Music, Stanford University, 2008

[8] O'Grady P. D. et Rickard S. T. *Automatic Hexaphonic Guitar Transcription Using Non-Negative Constraints*, Proceedings of the Irish Signal and Systems Conference, 2009

[9] Ozaslan T.H., Guaus E. *et al*. Attack Based Articulation Analysis of Nylon String Guitar, Proceedings of CMMR 2010, Malaga,

[10] Pettinen H. et Välimäki V. A time-domain approach to estimating the plucking point of guitar tones obtained with an under-saddle pickup. In Applied Acoustics, volume 65, pages 1207–1220, 2004.

[11] Puckette M. Patch for guitar. In Pd-convention, 2007. patches Puredata : http://crca.ucsd.edu/~msp/lac/.

[12] Rasamimanana N. *Geste instrumental du violoniste en situation de jeu: analyse et modélisation,* Thèse de doctorat, IRCAM/Université Paris 6, 2008.

[13] Reboursière L., Frisson C. *et al*. Multimodal guitar: A toolbox for augmented guitar performances. In Proceedings of NIME, 2010.

[14] Traube C. et Depalle P. Extraction of the excitation point location on a string using weighted least-square estimation of comb filter delay. In Proceedings of the Conference on Digital Audio Effects (DAFx), 2003.

[15] Wanderley M. M., *Interaction musicien-instrument : application au contrôle gestuel de la synthèse sonore*, Thèse de doctorat à l'Université de Paris 6, 2001.

PIANO TOUCH ANALYSIS: A MATLAB TOOLBOX FOR EXTRACTING PERFORMANCE DESCRIPTORS FROM HIGH-RESOLUTION KEYBOARD AND PEDALLING DATA

Michel Bernays
Université de Montréal
michel.bernays@umontreal.ca

Caroline Traube
Université de Montréal
caroline.traube@umontreal.ca

ABSTRACT

We hereby present the analytic tools developed as a MATLAB toolbox for exploring the most subtle features of piano touch and gesture. From high-sample-rate, high-precision precision key, hammer and pedal tracking data about a performance gathered thanks to the Bösendorfer grand piano-embedded CEUS digital recording system, the toolbox main functions can extract exhaustive features detailing the pianist's touch as a thorough account of nuances in articulation, timing, dynamics, attack and pedalling. Each performance and gestural control thereof is thus described over each of its notes and chords. By comparing several performances, it is possible to characterize the gestural control of expression in piano performance, as the correlations among piano touch features towards one examined expressive parameter. The analytic functions in the toolbox — with piano touch feature visualization and comparison, chords and notes selection tools, score-performance matching and advanced, automated statistical analyses and visualization thereof — allow for rigorous, quantitative exploration of expressive performance and its gestural control, which here is especially applied towards investigating the use of timbre as expressive device in piano performance. With these tools, we thus intend to build a gestural mapping of piano timbre.

1. INTRODUCTION

Musical performance is essential to the art and experience of music. Classical performers, in particular, aim to enlighten the composers' works, as reduced on scores, by their unique interpretation, thus expressing their creativity throughout this complex task. This holds undeniably true as regards piano performance, for which a vast repertoire has been composed in the few centuries since the inception of the instrument. An extensive, empiric knowledge of piano gesture and touch has thus developed within the pianistic world, to allow for the most vivid expressivity in piano performance. Many piano treatises, especially in the twentieth century [15] [18] [12], have provided guidelines for the most efficient and expressive gesture to improve the pianist sound. Through systematic studies and interview collections, researchers have highlighted the specificities of gesture, touch or fingering [3] strategies.

Moreover, along technological progress since the early twentieth century, new methods and tools have appeared that allow for a quantitative study of performance gesture.

The first datasets available on piano gesture in performances came as rolls recording key movements through mechanical design, such as Duo-art rolls which were used notably to study chord synchronization [27] and pedalling [11]. The Iowa Piano Camera, developed by Seashore et al. in 1936 for the "Objective Recording and Analysis of Musical Performance", could measure hammer motion and velocity with an elaborate, embedded slit-and-film system, and was used to analyze several dynamic, rhythmic/temporal and structural features [24]. Both systems could thus give out piano rolls indicating for each note the key depressed, its onset and offset (and thus duration), and in the last case, hammer velocity. In fact, this constituted a primitive version of the 1981 MIDI standard data. The subsequent MIDI recording pianos such as Yamaha Disklaviers, with the same but way improved pianoroll-like data acquisition abilities, vastly eased the quantitative recordings of some piano performance gestures. This way, systematic studies by Repp [22] [23], Parncutt [20] [21] and Goebl [5] [6] [7] among others, of such expressive features as timing, dynamics and articulation, have given us a better grasp the role of gesture in expressive piano performance.

Yet due to the limits inherent to the MIDI standard [17] with regard to the amount of information available on piano touch control, other technical methods of measurement have been called upon. In particular, and as early as 1929, Ortmann devised a mechanical recording system, involving springs, levers, a dynamograph and revolving drum, that could acquire continuous measurements of key motion and represent them as curves detailing the slightest fluctuations. It was thus used to study piano touch and tone [19] and determine the variations in touch, as key motion, for a single note with different tones intended. A few years later (1934), Hart, Fuller and Lusby [10] used an optical system to track hammer motion, and a mechanical key striker to identify the effect of various controlled key strokes on hammer motion and the acoustic signal.

Nowadays, equipment used to record expressive gesture in piano performance may include motion capture [26], sensors [9], or UV hand paint and lighting [14]. Such designs enable to acquire pianists' gesture as a whole, in

order to characterize musical embodiment or physiological features. As regards the instrumental gesture itself, that is the efficient part of performance gesture that conveys energy directly to the keys, recent recording pianos have expanded over MIDI limitations and propose highly accurate tracking of key (and pedal) depression and hammer movement. Such detailed information is actually required for analyzing the subtle nuances involved in some aspects of piano performance, among which timbre and tone control.

2. AIMS

As our main research project indeed aims at understanding high-level pianists' ability to control timbre and tone in their performances according to their musical and emotional intentions, we investigate how the most fine-grained properties of pianists' touch are nuanced within musical performances, in order to obtain different timbres. Yet such nuances necessitate highly precise data from which the intricacies of key strokes can be thoroughly assessed. Such is the data that can be acquired by the Bösendorfer CEUS piano digital recording system. In order to analyze this fine-grained data, we have aimed to develop a dedicated MATLAB toolbox including a set of functions that we are hereby describing in this paper.

3. HIGH-RESOLUTION DATA ACQUISITION

The CEUS system we used is embedded in an Imperial Bösendorfer grand piano (Figure 1). [1] The system includes optical sensors behind the keys, hammers and pedals, microprocessors and electronic boards (Figure 2) that process sensor data and send it to an embedded computer into whose hard drive data is stored. CEUS is also a reproducing system, with solenoids attached to each key for replicating the exact motion stored from a human performance. The system tracks keys and pedals position (via their angle relative to rest level) at a standard rate of 500 Hz, and over 250 steps (8-bit encoding), which roughly means for keys a $40\mu m$ tracking accuracy for the point of the key (or about 30" for the key angle). As for hammers, two sensors monitoring the hammer ballistic travel towards the string give out its maximum velocity after it is launched by the key.

These datasets are recorded as binary files, wherein successive data chunks correspond to each timestamp — one every two milliseconds. Each chunk starts with a break code (255), followed by a 24-bit value indicating milliseconds since the recording start. It is followed by a series of 16-bit blocks, one for each key, pedal or hammer activated at this timestamp. Each block first contains the 8-bit number of the key depressed (as in MIDI, with the central C4 equal to 64, and up to 108), or pedal (109 to 111), or hammer (MIDI-key number + 128). The second 8-bit number gives out key/pedal position or hammer velocity. Datasets are stored under the `.boe` extension format.

[1] This piano is installed in a dedicated studio at BRAMS (International laboratory for Brain, Music and Sound research), *Université de Montréal.* (http://www.brams.org)

Figure 1. Imperial Bösendorfer grand piano in the BRAMS studio, with embedded CEUS system.

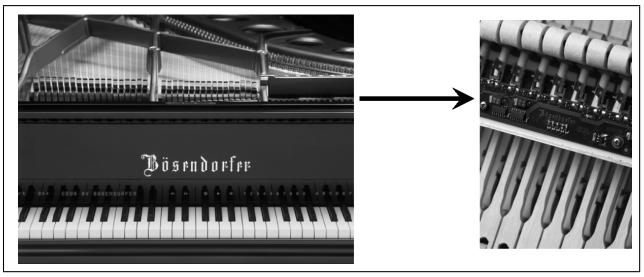


Figure 2. Details of the CEUS system: fallboard display/ interface and embedded electronics.

The Bösendorfer CEUS recording system thus constitutes an extremely precise tool to observe the finest subtleties in pianists' touch and measure the part of gesture actually efficient and transferred to the piano action. However, in order to get a clear understanding of piano touch peculiarities involved in one performance, the raw data must be processed in a more intelligible way. This is the role of the primary functions in our toolbox.

4. DATA PROCESSING

4.1. From streamed data files to piano rolls

The first requirement in the chain processing that would lead to a thorough *a posteriori* analysis of boe files, is to parse the raw files into MATLAB matrices. Binary-to-decimal conversion is handled by MATLAB, and our function `extract.m` deals with identifying each data chunk relative to one timestamp, essentially by finding the break tags "255", then storing timestamp-and-value (of key or pedal depression or hammer velocity) data couples on one line. The parser can also deal with the older boe format, wherein data is encoded in ASCII with hexadecimal numbers. We thus get a streamline of events that happened in the recorded performance.

Yet data is easier to use and interpret once restructured

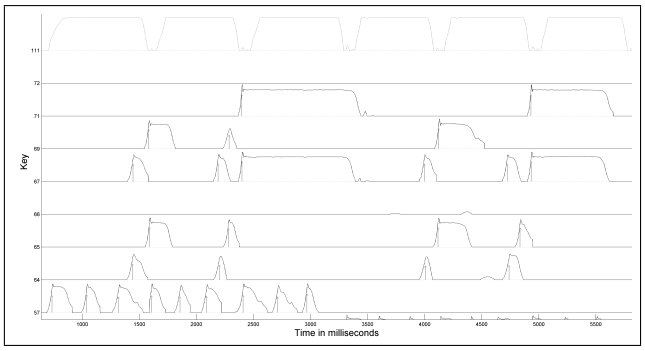


Figure 3. Pianoroll display of a performance (detail), wherein key motion is shown as blue lines, MHVs as red vertical arrows, and sustain pedal in light blue.

in a key-by-events matrix, wherein each line corresponds to one key/pedal/hammer, with its MIDI-like number as referent, and successive blocks indicate each event with its timestamp and value. From a timestamp-driven structure, the `notetime.m` function thus reorganises data into a key-by-key account of events. Special attention is given to correcting for possibly missing information, such as one key being depressed at two timestamps t and t+4 yet missing information at t+2. In all, missing or redundant timestamps are filtered, as well as singleton events. [2] This structured data is exported into a text file.

The data format is especially useful to get a visual representation of the recorded performance, in the form of a piano roll. To this aim, the `pianoroll.m` function can display keys and pedals motion, one line for each, along time. All recorded maximum hammer velocities are displayed for their relative key, with a red vertical arrow (Figure 3). This gives out the graphic equivalent of the well-known MIDI pianoroll display, but with the exact level of key depression instead of a fixed-velocity block per note.

However, within this data structure and pianoroll display, we can only observe the high-precision linear response from each key motion, and do not have direct information on the most basic musical structure, that is, the note. While visually obvious, this basic information we get immediately in MIDI here has to be retrieved through additional processing.

4.2. Retrieving notes

As events (key angle) are only registered when the key is out of its rest position, we can essentially retrieve note onsets by finding discontinuities in the timestamp sequence related to one key. Note onsets thus occur whenever two consecutive, timestamp-and-value blocks stored in "notetime" files possess non-consecutive timestamp values (*i.e.* separated by more than 2 ms). This process is run in the `notes.m` function, while improved with correction procedures for missing timestamps (up to 8 ms deep) and noisy information at note onset — *e.g.* a key somewhat being pushed down of less than a millimeter when the pianist sets his finger on top of it in preparation for the next note — which could shift onset time towards way earlier than actually performed. Retrieved notes that prove too feeble to launch the hammer and produce a sound are also filtered out. Moreover, the note-detection function can account for successive notes played on the same key, with the second note starting before the key is fully released from the first. Indeed, the double escapement action featured on grand pianos allows for faster note repetition after just a partial key release, down to a threshold called escapement point. As such two notes are not separated in the data by a discontinuity in timestamp sequence, we used a forward-loop exploration of local minima. The escapement threshold was assessed empirically through performance corpus analysis, and could be set at 140 (upon 250). Local minima lower than this threshold define the onset of a new note.

Once the notes thus identified, we set on exploring the upper-level musical structure: chords.

4.3. Identifying chords

The function `chords.m` identifies groups of notes which have near-synchronous onsets. The task is performed in two rounds. First, a group of notes is formed when their onsets are less than 50 ms apart. A chord is thus defined, whose onset is defined as the earliest note onset. Another note can then be assigned to this chord if it provides the best-fitting, under-50-ms onset timing difference. This ensures the most synchronous notes are grouped together. In the second round, chords can be merged if any of the note onsets (instead of the earliest) within one chord falls within a looser interval, 100 ms, of any of the note onsets from other chords. The 50 and 100 ms onset synchronism intervals are consistent with mean and upper-bound values found by Repp [22] and Shaffer [25] and were tested empirically for matching between the chords identified in a corpus of performances and the designations in their respective scores.

Additionally, in the context of our target experiment which involved four different pieces, we were able to set thresholds for separating the range of each hand — that is, a note above which the left hand, and below which the right hand, never play. Chords could thus be separated by hand.

From the high-precision response tracking of key and pedal depressions and hammer velocities, we thus retrieve the fundamental musical structure of notes and chords. [3] This allows us to get back up to MIDI-like note identification, add the definition of chords, and deal with the fine-grained information through reduction to exhaustive and relevant features set to describe each note and chord.

[2] This procedure has essentially become a precautionary tale, as with the latest CEUS software updates such acquisition errors have been all but eliminated.

[3] The notion of "chord" is hereby used loosely, as it can account for a single note when not deemed synchronous to any other note played by the same hand.

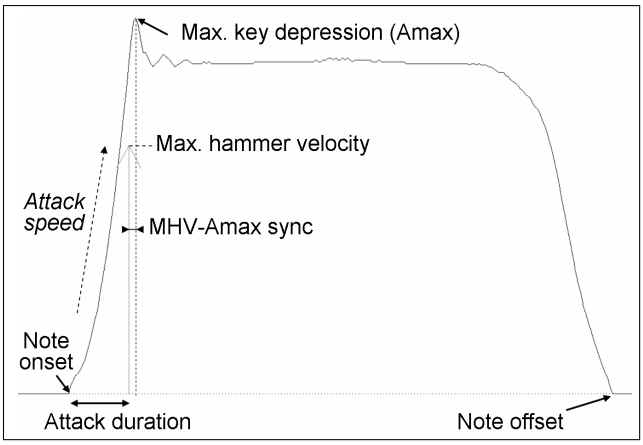


Figure 4. Illustration of some essential note features.

5. PIANO TOUCH DESCRIPTIVE FEATURES

Our toolbox also provides a thorough analysis of notes and chords, and extracts numerous features from the high-precision, 500 Hz tracking of keys and pedals depression and maximum hammer velocities record. The exhaustive information related to each musical structure of note or chord is reduced to a large set of features most relevant to understanding piano touch. The most relevant features to set alight were chosen on the basis of studies of various aspects of expressive piano performance that used MIDI data [22] [23] [2] [6], as well as studies focused on piano touch [19] [21] or keyboard action [16]. We have programmed, adapted, extended or added descriptive features which can be sorted in several broad categories: dynamic level, attack speed and type, duration and sustain, release, synchronism, intervals and overlaps, and pedals use.

5.1. Single note features

Each note is individually described by 46 features. First are its basic characteristics (Figure 4): key number, onset, offset and duration; maximum hammer velocity (MHV), maximum key depression angle (Amax) and their corresponding timestamps. From these are calculated several attack-descriptive features: attack duration (related to instants of both Amax and MHV), attack speed (as a ratio of Amax or MHV to its duration), and timing between Amax and MHV. Those features assess the attack as a description of dynamic level — the faster the attack and the more synchronous Amax and MHV, the higher dynamic level is.

Attack type can also be defined as percussive or not. Indeed, as Goebl et al. have shown [6] key trajectory differs whether the key is pressed, with the finger initially resting on the surface of the key, or struck percussively, with the finger starting above the key and reaching it with a non-zero velocity. The struck touch displays a fast initial key depression that slows down until reaching (or not) the keybed, while a pressed touch has the key depressed slowly first and gradually accelerating. We used two methods to elicit this behaviour: the ratio of key depression at half the attack duration to the maximum key depression, and the mean key depression during attack — akin to the area swept by the key depression curve during attack. Both features will have higher values with a percussive touch.

Two other ways of assessing the note profile were designed. First, critical points akin to the acoustic temporal envelope were retrieved, thus defining up to four zones in key depression: attack, decay (a short drop up in key depression certainly due to the reaction of the keybed felt, and found in many notes), sustain and release. Attack, sustain and release durations and ratio to the total note duration were thus assessed. And second, we defined empirically a threshold over which the key can be deemed deeply depressed. From there we could define three sections, and their durations, the first before the key reaches the threshold, then while the key is depressed over the threshold, and when it falls below it — akin in most cases to attack, sustain and release respectively.

Finally, we gathered sustain and soft pedals use during the note: for each, their duration of use and amount of depression during the note, as well as their depression at note onset, offset and at the instant of MHV.

5.2. Chord features

Each chord is first described by basic features: number of notes within, its onset and offset (earliest and latest of its note onsets and offsets resp.), duration and maximum of its note Amax and MHV. Then, each of its notes is assigned, besides its 46 individual features, 10 additional characteristics of its synchronism towards the chord: note onset lag on chord onset, its ratio to the chord duration, and the onset lag amount (defined as the sum of all other keys depressions within the chord before the onset of the target note); note offset lead, ratio and amount with regard to chord offset (defined in the same way); its synchronism with the chord as ratio of note duration to chord duration; and its synchronism amount, defined as the ratio of the total amount from which the other notes within the chord are depressed for the duration of the target note, compared with their total depression amount. Each note within a chord is thus described by 56 features.

And each chord in itself is also described by 56 other features, with descriptions of internal synchronism and pedal use in addition to the primary features explained above. Indeed, the smallest, non-zero note onset lag defines the chord melody lead (how early the first note is compared with the next), and the smallest, non-zero note offset lead sets the one-note trail within the chord. As for soft and sustain pedals use, both are exhaustively featured as follow: duration and amount of depression during the chord; duration of deep-depression and mid-depression (when pedal depression level falls above or between certain thresholds, resp.); and assessment at chord onset and offset of pedal depression levels, activation (on or off) and timing (how long before or after the pedal was or will be activated).

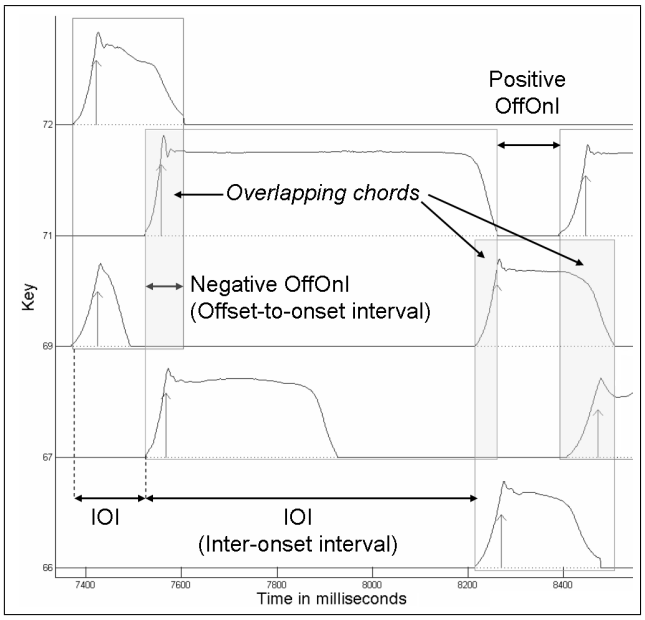


Figure 5. Pianoroll display of successive (framed) chords with their interval and overlap relations pointed out.

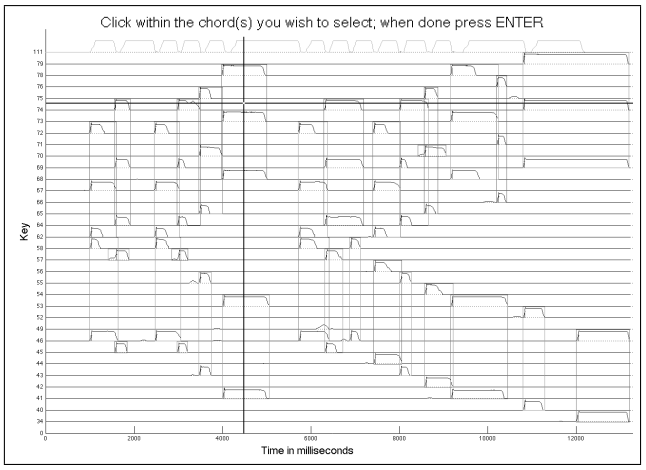


Figure 6. Chord selection interface, wherein a black target cursor (here centered around key 74 and timestamp 4500) indicates which chord a click within the frame will select.

Finally, the following features assess the relations between chords. First, intervals are measured: inter-onset interval (IOI) from one chord onset to the next, interval from chord offset to the next chord onset (OffOnI), and its direction — negative indicates legato, positive staccato. Then, overlaps between chords are defined through their duration (how long one chord is overlapped by others), amount (of depression) and number (of chords in overlap with the target) (Figure 5). All those features of intervals and overlaps are calculated with regard to any chord (in onset temporal order), to same-hand chords only, and to chords played the other hand only. This is meant to elicit different articulation strategies.

In all, each multi-note chord is described by 56 chord-specific features, plus the mean and standard deviation of the 56 features describing each of its notes. This thus amounts to 168 features per chord. [4] Such an exhaustive account clearly states the depths at which the CEUS system lets us observe piano performance, and gather quantitative information about piano touch and its dynamics, percussiveness, articulation, depth, timing, pedalling,etc.

This features extraction is processed within the functions `notes.m` and `chords.m` (the latter calling the former), and is output in a three-dimensional, chord-by-note-by-feature matrix. For each chord of n notes, there are (n+3) vectors, the first being the features describing the chord itself, the second the mean of its note features and the third their standard deviation. Additionally, the mean and standard deviation of all chord features gives out a description of the performance, as a whole, through 322 characteristics. Moreover, each performance is similarly described with regard to left-hand chords only, right-hand chords only and left-vs-right hand differences. The results matrix can also be printed out as a formatted text file (restructured in 2D), or as two separate files accounting for chords alone and notes alone, resp.

[4] In the case of a single-note "chord", due to redundancy and no standard deviation, there remains 85 valid descriptors.

6. ADDITIONAL FUNCTIONS

In addition to the main chord-and-note structuring and feature extracting functions, the toolbox also contains several analysis tools.

6.1. Selection

First, subsets of chords or notes can be selected with the `select_chords.m` or `select_notes.m` functions. There are two ways of selecting the subset: either by indicating notes or chords to select in a matrix or text file called as input argument, or graphically, by clicking on the chords/notes one wishes to select on the pianoroll display of the performance. With the first method, most useful for batch processing, notes to select are to be specified, each on one line, by their key number and timestamp falling between their onset and offset. The same is asked for chords to select, and here any key number falling within the range between the lowest and highest notes in the chord can be used as referent. In case such coordinates could refer to more than one note or chord (due to overlap), the closest fit in time range is selected. The graphical counterpart method works essentially the same way, with each click on the note or chord to select within the pianoroll figure sending back a key number and timestamp. Chord selection by click is eased by the framing of all chords in the pianoroll figure (Figure 6). The function then retrieves the features describing each selected note or chord, and calculates means and standard deviations over the subset. It outputs a 3D results matrix similar to the main performance-features matrix previously described, with the same printout options. This way, one can assess local characteristics within a performance.

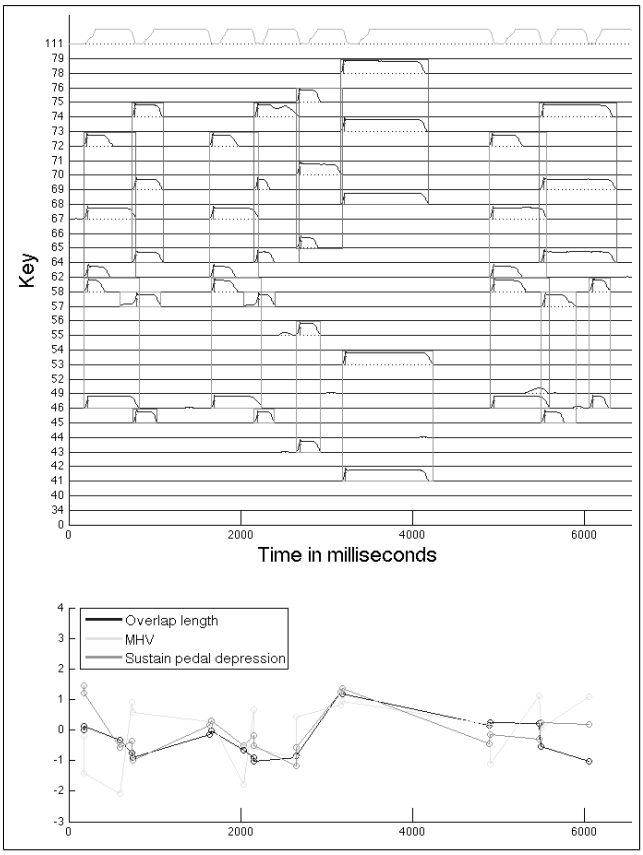


Figure 7. Graphical display of one performance piano roll (detail) and evolution over time of selected features.

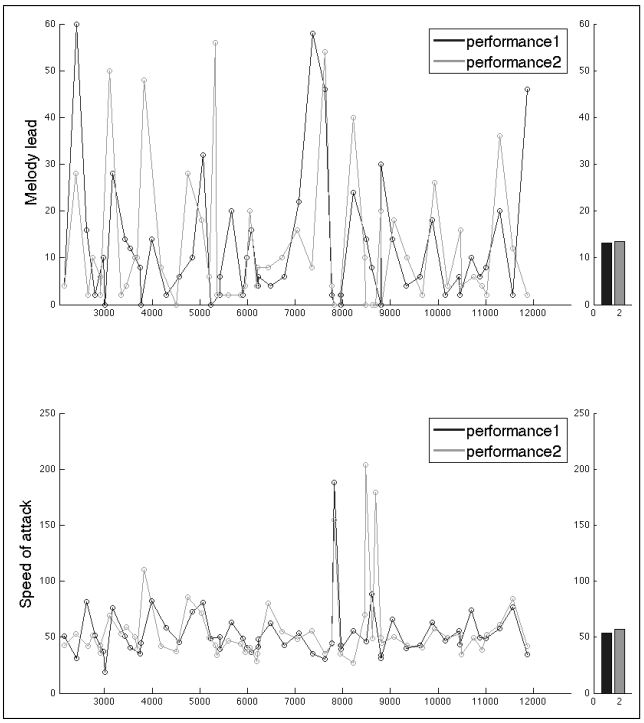


Figure 8. Comparison in time of 2 performances for 2 features, and mean thereof.

6.2. Graphical feature representation

In order to visualize the evolution of features over the duration of a performance, the `g_pianostats.m` function plots the normalized feature values against the performance pianoroll display. First, a graphical user interface allows to select the features to plot and specify some options (hand separation, error bars for note features). The value of each selected feature, for each chord, is then plotted over time at the instant of chord onset (Figure 7). With the pianoroll as reference, one thus can see the evolution of the feature along the performance, and possibly identify its relation to the musical structure — *e.g.* phrasing — of the piece.

The second function, `g_compare.m`, is a graphical comparison tool, in time, of several performances. The same graphical user interface is used to select the features to display. For each selected feature, a separate plot tracks its evolution in time within each input performance, and compares its mean value within each performance. This allows to identify what differs between performances, and especially when differences occur (Figure 8).

6.3. Score matching

This function is aimed to assess the fit between two performances or between a performance and its score. It compares the notes identified in one performance to another, or to the score (rendered as MIDI). The function only compares MIDI-like information of key number, onset, duration and velocity (MHV). It is an adaptation, with our custom, derived CEUS-data complying parser, of Ed Large's score matching functions [5] [13], essentially intended to assess performance errors (such as missing notes).

The function returns a graphical display of two corresponding MIDI piano rolls, wherein each note within one performance is linked to its corresponding note in the other performance/score. The cross-correlation matrix of the two performances/score in time is also displayed, and we added the calculation of a matching rate, which indicates the percentage of notes that match (in key number, timing, duration and velocity) between the two performances.

6.4. MIDI to boe conversion

Last but not least, we designed the `boe_gen.m` function to generate CEUS boe format files from MIDI input. First, the MIDI files are parsed into MATLAB variables with Ken Schutte and Tuomas Eerola `midi2nmat.m` function, from which output we conserve each note with its key number, onset, duration and velocity. MIDI velocity (7-bit) is linearly converted to CEUS MHV (8-bit). Our function can also directly take text files as input that use the same format as midi2nmat output. Instant of MHV can also be specified as a fifth parameter. If not (or MIDI input), instants of MHV are extrapolated as an empiric function of MHV and duration.

[5] Available online at https://www.jyu.fi/hum/laitokset/musiikki/en/...research/coe/materials/miditoolbox/matchingPerformancetoNotation.zip

Then, boe files can be generated with a straight rendering of the input information, that is, with MIDI-style notes of constant MHV/velocity along their duration, and a crenelated outlook. Yet the function can also generate more realistic boe files wherein key depression patterns are rendered as if key motion were precisely tracked. Depending on the input parameters, each note is assigned one of three fine-grained note prototypes each representing a typical key depression profile — as commonly identified in CEUS boe recordings. The note prototype is then warped by polynomial interpolation so as to fit the note input parameters, with longer durations accounted for by stretching the sustain phase (thus keeping attack and release in valid forms). The key-by-events "notetime" matrix is first created, and then transcribed in the output boe file.

This MIDI (or MIDI-like) conversion and/or augmentation into the CEUS system boe format has proven useful for testing the feature-extracting functions, and for comparing performances, feature-wise, to the "flat" reference of their MIDI-rendered scores.

7. APPLICATION: ANALYSIS OF PIANO TIMBRE GESTURAL CONTROL

7.1. Context

Amongst the various aspects of expressivity in piano performance, our research project focuses on timbre, and the control thereof through subtle nuances of touch and applied gesture. In order to investigate this timbre and tone dimension so widely acknowledged within the pianistic community as central to the art of piano playing [18] [4], we designed an experiment set in a musically relevant context. This has let us explore farther than the single-note studies which helped reinforce the long-held notion within the scientific world that mechanical constraints limited piano timbre control, for a single, isolated piano key, to the sole intensity of key stroke [10]. The study thus follows the groundbreaking, yet somehow overlooked, work of Ortmann [19], who could identify different patterns of key depression depending on tone intentions. He also posited that tone combinations and blends, when several notes are played simultaneously or successively and pedals are used — as is the case in musical context — add so many possibilities and factors of tone control [8]. We thus explored how features of articulation, timing, dynamics, attack, touch percussiveness, pedalling, and so on are nuanced in musical performances in order to produce different tones and timbres.

7.2. Method

To this aim, we first identified five timbre descriptors most salient and representative of the piano timbre range [1]: *Dry*, *Bright*, *Round*, *Velvety* and *Dark*. Four miniature pieces were composed so that each could be fittingly performed with each of the five timbres. And with the CEUS data acquisition system, we could record performances by four pianists with professional experience. They each played each piece, tree times with each of the five timbres. We thus compiled 240 boe recordings, from which we extracted the exhaustive characteristic features thanks to our Bosen Toolbox.

7.3. Analysis

We then set to compare all performances through their mean features, with regard to their respective timbre. To this aim, we developed several analysis functions as an extension of the Bosen Toolbox. These functions allow for automated processing of all features through several statistical tests from MATLAB Statistics Toolbox.

The main function performs several variance tests over each feature, with timbre as factor. The three tests used are one-way ANOVA, Welch robust test of equality of means, and Kruskal-Wallis non-parametric rank analysis of variance, with the assumptions required for test validity (normality of same-timbre groups distribution for ANOVA and Welch, and Levene's homogeneity of variance for ANOVA) tested as well. For each feature, with the p-value of the most powerful test whose assumptions are not violated as indicator of significance at the .05 level, we thus know whether this feature significantly varies depending on timbre, and therefore whether the feature can be useful in determining the timbre performed. If so, post-hoc tests (with Tukey's honestly significant difference criterion) are run to determine the timbre pair-wise significance for the feature, that is, which two timbres among all pair combinations the feature values can set apart from one another. All those results, for all features, are returned in a table.

In addition, the normalized means and standard deviations over all performances of same timbre are calculated, for each timbre and each feature. Those values can be graphically compared with a linear plotting function. Moreover, normalized feature values, regrouped by timbre, can be directly and more thoroughly displayed as box plots. Finally, the same information can be represented as a Kiviat graph — a.k.a. "radar" or "cobweb" chart.

Furthermore, a function was designed to perform Principal Component Analysis over all significant features. PCA identifies, one by one, the linear combination of features (with individual weights assigned) which can explain as much of the total variance (or of the remaining variance) as possible. Within the space thus defined by those dimensions, each performance possesses its own coordinates. Our function selects all dimensions that explain a large enough chunk of the total variance (over 10%), and traces all the plans formed by the remaining dimensions, with the performances (color-tagged for timbre) set therein according to their coordinates. The description of those dimensions as linear combinations of the significant features is also stored. This method thus gives a rapid way to identify which combination of features is best able to differentiate between timbres.

7.4. Results

All these methods and processes were applied to the performances recorded for the experiment and their gestural descriptors extracted with the Bosen Toolbox. We separately analyzed the whole set of descriptors, and their means and deviations per performance, for full performances, for their left- or right-hand chords only, and for performances grouped piece by piece and pianist by pianist. While the complete interpretation of these exhaustive analyses is still currently under way, we have already identified significant correlations of several gesture descriptors to the timbre performed. The eight most representative among all significant timbre-discriminating gestural features are displayed as a Kiviat graph (Figure 9). Thus are elicited the different strategies in dynamics, pedalling, articulation, *rubato* and touch employed to perform a certain timbre. We can thus gather how piano touch and gesture features were used to control and vary timbre nuances. For instance, a counter-clockwise account of the Kiviat graph, starting to the right, indicates that playing with a *velvety* timbre required low yet quite varying dynamics, variation in attacks, heavy use of the soft pedal and to a lesser extent of the sustain pedal, a lot of *legato*, rather stable chord durations (*non rubato)* and a very soft, non-percussive touch. By comparison, playing *dark*, while most similar to playing *velvety* according to those eight gesture descriptors, involves a little more dynamic power and less variations, while speeds of attack still vary greatly. [6] The sustain pedal is used more, and the soft pedal less so. The articulation is even more *legato*, while the *dark* touch is much more percussive than the *velvety* touch.

The results thus consist in effect in the gestural mapping of piano timbre, to the extent of the gesture descriptors extracted and the timbres considered. The coarse timbre mapping with only these eight select features still allows for a unique description of the five timbres. More thorough analyses, involving more gesture descriptors for a finer account and interpretation of timbre gestural mapping, are in the works.

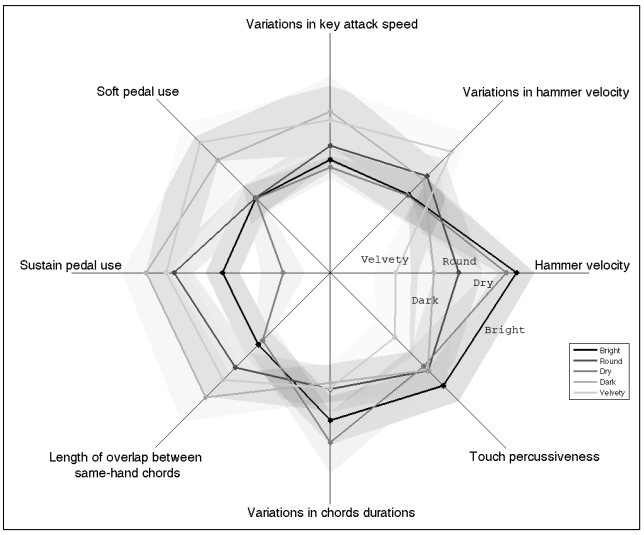


Figure 9. Kiviat plot of the eight most representative timbre-discriminating gesture descriptors (shaded zones correspond to $\pm$ 2 S.E.).

8. DISCUSSION

The Bosen Toolbox was developed to make the most use of the high-accuracy, high sample-rate Bösendorfer CEUS key/hammer/pedal tracking system data, and to offer an exhaustive and thorough account of piano touch and gesture, through meaningful features that can be interpreted in a musically relevant way.

Moreover, the toolbox statistical functions allow for analysis automation, and can be easily adapted to studying whichever performance factor, with a single configuration file to define all variables.

And, while the toolbox was especially designed for the CEUS data format, it could conceivably be used with any similar high-precision equipment, capturing the keyboard discretized per key (up to 108 discrete units) on one dimension, and high-accuracy key depression tracking on another. For instance, a high-frame-rate video tracking system could be positioned in such a way that its field of view would encompass a whole keyboard and key depressions would be visible (*e.g.* with the camera set at one end of the keyboard, or somewhere below it). The same would hold true for a motion capture system with reflectors installed on each key. More creative designs could also be conceived that do not even involve a keyboard...

To those ends, we plan on making the Bosen Toolbox, once finalized, available online — as a set of MATLAB m-files — under GNU licence. For now, the toolbox can be sent upon request.

In conclusion, the Bosen Toolbox is meant to help better understand the subtle nuances in expressive musical performance through which artists masterfully manage to convey emotion and feeling.

9. ACKNOWLEDGEMENTS

We wish to thank all the pianists who participated in the study. This research is supported by FQRSC (*Fonds de recherche du Québec - Société et culture*) through a doctoral research grant. Additional financial support was provided by CIRMMT (Centre for Interdisciplinary Research in Music Media and Technology) and OICRM (*Observatoire interdisciplinaire de création et de recherche en musique*). Research equipment, most crucially the Bösendorfer CEUS digital recording grand piano, is provided by BRAMS (International Laboratory for Brain, Music and Sound Research).

[6] Variations in hammer and key attack velocities resp. may not be linearly correlated, although they both concern the same broad-sense action, because they can be affected differently by differences in dynamic register and touch percussiveness.

10. REFERENCES

[1] Bernays, M. & Traube, C. "Verbal expression of piano timbre: Multidimensional semantic space of adjectival descriptors", in A. Williamon, D. Edwards & L. Bartel (Eds.), *Proceedings of the International Symposium on Performance Science (ISPS2011)*, Toronto, ON. European Association of Conservatoires (AEC), Utrecht, Netherlands, pp. 299–304, 2011.

[2] Bresin, R. & Battel, G.U. "Articulation strategies in expressive piano performance", *Journal of New Music Research*, 29(3): 211–224, 2000.

[3] Clarke, E.F., Parncutt, R., Raekallio, M. & Sloboda, J.A. "Talking fingers: an interview study of pianists' views on fingerings", *Musicae Scientiae*, I(1): 87–107, 1997.

[4] Dawei, Y. "Melody and timbre in piano performance", *Canadian Social Science*, 2(1): 69–72, 2006.

[5] Goebl, W. *The Role of Timing and Intensity in the Production and Perception of Melody in Expressive Piano Performance*. PhD thesis, *Institut für Musikwissenschaft, Karl-Franzens-Universität*, Graz, Austria, 2003.

[6] Goebl, W., Bresin, R. & Galembo, A. "Touch and temporal behavior of grand piano actions", *Journal of the Acoustical Society of America*, 118(2): 1154–1165, 2005.

[7] Goebl, W., Flossmann, S. & Widmer, G. "Investigations into between-hand synchronization in Magaloff's Chopin", *Computer Music Journal*, 34(3): 35–44, 2010.

[8] Gustafson, A.E. *Tone production on the piano: the research of Otto Rudolph Ortmann'*. DMus thesis, University of Texas at Austin, 2007.

[9] Hadjakos, A., Aitenbichler, E. & Mülhäuser, M. "Potential use of inertial measurement sensors for piano teaching systems: motion analysis of piano playing patterns", *Proceedings of the 4th i-Maestro Workshop on Technology-Enhanced Music Education*, Genova, Italy, pp. 61–68, 2008.

[10] Hart, H.C., Fuller, M.W. & Lusby, W.S. "A precision study of piano touch and tone", *Journal of the Acoustical Society of America*, VI: 80–94, 1934.

[11] Heinlein, C.P. "A discussion of the nature of pianoforte damper-pedalling together with an experimental study of some individual differences in pedal performance", *Journal of General Psychology*, 2: 489–508, 1929.

[12] Kochevitsky, G. *The Art of Piano Playing: A Scientific Approach*. Summy-Birchard Music, Secaucus NJ, 1967.

[13] Large, E.W. "Dynamic programming for the analysis of serial behaviors", *Behavior Research Methods*, 25(2): 238–241, 1993.

[14] MacRitchie, J. *Elucidating Musical Structure through Empirical Measurement of Performance Parameters*. PhD thesis, University of Glasgow, UK, 2011.

[15] Matthay, T. *The Visible and Invisible in Pianoforte Technique*. Oxford University Press, 1932.

[16] McPherson, A. & Kim, Y. "Multidimensional gesture sensing at the piano keyboard", *Proceedings of the 29th ACM Conference on Human Factors in Computing Systems (CHI 2011)*, Vancouver, B.C., pp. 2789–2798, 2011.

[17] Moore, F.R. "The dysfunctions of MIDI", *Computer Music Journal*, 12(1): 19–28, 1988.

[18] Neuhaus, H. *The Art of Piano Playing*. Trans. from original edition (1958) by K.A. Leibovitch, Barrie & Jenkins, London, 1973.

[19] Ortmann, O.R. *The Physiological Mechanics of Piano Technique: An Experimental Study of the Nature of Muscular Action as Used in Piano Playing and of the Effects Thereof Upon the Piano Key and the Piano Tone*. Reprint of original edition (1929) by E.P. Dutton, New York, 1962.

[20] Parncutt, R., Sloboda, J.A. & Clarke, E.F. "Interdependence of right and left hands in sight-read, written, and rehearsed fingerings of parallel melodic piano music", *Australian Journal of Psychology*, 51(3): 204–210, 1999.

[21] Parncutt, R. & Troup, M. "Piano", in R. Parncutt & G. McPherson (Eds.), *The Science and Psychology of Music Performance: Creative Strategies for Teaching and Learning*, Oxford University Press, pp. 285–302, 2002.

[22] Repp, B.H. "Patterns of note onset asynchronies in expressive piano performance", *Journal of the Acoustical Society of America*, 100(6): 3917–3932, 1996.

[23] Repp, B.H. "Acoustics, perception, and production of legato articulation on a computer-controlled grand piano", *Journal of the Acoustical Society of America*, 102(3): 1878–1890, 1997.

[24] Seashore, C.E. *Objective Analysis of Music Performance*. University of Iowa Press, 1936.

[25] Shaffer, L.H. "Performances of Chopin, Bach, and Bartók: Studies in motor programming", *Cognitive Psychology*, 13: 326–376, 1981.

[26] Thompson, M.R. & Luck, G. "Effect of pianists' expressive intention on amount and type of body movement", *Proceedings of the 10th International Conference on Music Perception and Cognition (ICMPC10)*, Sapporo, Japan, pp. 540–544, 2008.

[27] Vernon, L.N. "Synchronization of chords in artistic piano music", in C.E. Seachore (Ed.), *Objective Analysis of Music Performance*, University of Iowa Press, pp. 307–345, 1936.

LOGICIEL D'AIDE A LA CONFIGURATION D'UN SYSTEME DE SPATIALISATION SUR ACOUSMONIUM

Rudi GIOT
LARAS - ISIB
giot@isib.be

Alexis BOILLEY
Musiques et Recherches
technique@musiques-recherches.be

Ludovic LAFFINEUR
LARAS - ISIB
ludovic.laffineur@gmail.com

RÉSUMÉ

La mise en place d'un acousmonium dans un lieu de spectacle et sa configuration sont des processus qui consomment beaucoup de temps. Les salles de spectacle ont un coût de location souvent élevé et sont donc réservées pour un nombre de jours limité. Souvent, à l'installation du système, les réglages s'éternisent et font perdre de précieuses minutes pour les répétitions. Nous proposons donc un logiciel qui permet de préparer le travail avant le concert et d'accélérer les derniers réglages du système lors de sa mise en place. Notre application permet de planifier l'implantation des haut-parleurs, de réaliser des configurations de base en fonction des différents formats des oeuvres jouées au concert, d'imprimer les fiches techniques, les plans de câblage et de disposition des diffuseurs. Pendant les répétitions, elle permet d'avoir une vue synoptique de la salle et de présenter la console de mixage aux interprètes de manière plus intuitive. Elle leur donne aussi la possibilité d'adapter la configuration en fonction de leurs désidératas. L'application est prévue pour être contrôlée à la souris mais aussi par une surface multi-touches. Cette dernière apporte une ergonomie plus importante et ouvre des perspectives nouvelles dans le domaine de la spatialisation pendant le concert.

1. INTRODUCTION

La mise en espace de musiques acousmatiques en concert est un problème relativement complexe. L'acousmonium, l'instrument de diffusion, est composé d'un ensemble de haut-parleurs de couleurs sonores différentes placés autour du public. Il est généralement contrôlé par une console de mixage permettant à l'interprète la gestion de la spatialisation de l'oeuvre. Le nombre d'enceintes étant souvent important, le câblage est laborieux, la configuration non conventionnelle de la console de mixage est complexe et la préparation de l'installation sur papier induit des risques d'erreurs. Il est donc nécessaire de respecter un ensemble de bonnes pratiques garantissant la qualité de la diffusion, de la mise en place du dispositif jusqu'à son utilisation. Ces bonnes pratiques, appliquées à la lettre, consomment de longues heures qui sont ponctionnées sur les temps de répétitions. Pour réduire les coûts de chaque concert et donner plus de temps de préparation aux interprètes, il est important d'optimiser le temps de mise en place et de réglage du système.

Avant notre projet, la configuration du système était relativement rigide. Il était difficile, voire impossible, de modifier l'assignation des curseurs de la table de mixage. De plus, les documents d'installation du matériel n'étaient pas normalisés tant au niveau des techniciens et régisseurs qu'au niveau des interprètes.

Il était donc urgent de réaliser un logiciel permettant la préparation et la gestion de l'installation d'un tel système en répondant aux problématiques énoncées.

2. ARCHITECTURE

Le déploiement de l'application se fait pour l'instant sur deux machines. La première gère l'interface, et la seconde gère la partie son.

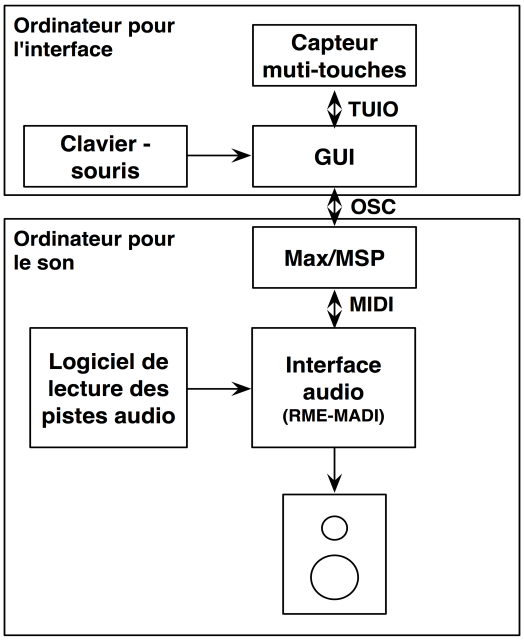


Figure 1. Architecture de l'application.

Comme on peut le voir sur la figure 1, le système est divisé en quatre parties :

- le GUI (Graphical User Interface) présente les écrans de configuration et envoie les commandes vers Max/MSP [11]
- le patch Max/MSP établit le pont entre la commande et la gestion de l'interface audio

- l'interface audio réalise le matriçage, le routage et le mixage de l'audio
- le logiciel de lecture lit les pistes audio des œuvres du concert

L'interface graphique est programmée en Processing [16]. Cet environnement est choisi pour son aspect multiplateformes et ses qualités au niveau du prototypage rapide.

Pour des considérations ergonomiques, nous avons choisi de travailler avec une surface multi-touches. Ce capteur communique avec notre logiciel via le protocole TUIO [17]. Ce protocole est intéressant dans la mesure où il est ouvert et indépendant du matériel. Nous garantissons ainsi le portage de l'application sur une autre surface multi-touches ou sur un autre système d'exploitation. Le programme est réalisé de manière à être également utilisable sur une machine conventionnelle, sans multi-touches, avec une souris et un clavier.

Les réglages réalisés au niveau du GUI sont envoyés à Max/MSP à travers des messages OSC [15] sur UDP/IP/Ethernet. Ce protocole est standard, ouvert et multiplateformes.

La couche logicielle introduite dans Max/MSP est nécessaire pour garantir la compatibilité du GUI avec des interfaces audio différentes. Les utilisateurs ont donc un libre choix au niveau de leur interface audio-numérique (une carte MADI [18] est utilisée dans notre configuration).

Le logiciel de lecture audio, Digital Performer dans notre cas, peut directement envoyer le signal audio dans les convertisseurs de la carte-son qui aiguille les flux de données et ajuste les différents niveaux dans sa matrice. Nous garantissons ainsi une fiabilité et une qualité de mixage du signal qui dépend uniquement des capacités de l'interface audio-numérique choisie.

3. LE LOGICIEL DE CONFIGURATION

Nous avons séparé la configuration du système en différentes étapes représentant chacune une séquence nécessaire à l'accomplissement de la tâche suivante.

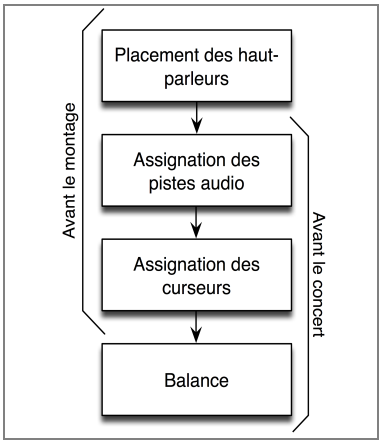


Figure 2. Les différentes couches de l'application.

Nous avons ainsi identifié quatre couches logicielles (figure 2) interdépendantes qui se présentent sous forme de quatre onglets dans l'interface graphique.

Nos quatre couches peuvent être utilisées à deux moments distincts :

- avant le montage: on réalise la conception, l'édition des documents techniques nécessaires à l'installation et la pré-configuration des assignations
- avant le concert: on calibre le système et on crée les configurations personnalisées pour chaque oeuvre répondant aux besoins de chaque interprète ou aux consignes du compositeur

3.1. Placement des haut-parleurs

Dans cette première étape, on affiche (figure 3) dans la partie gauche de l'écran un plan épuré de la salle dans laquelle le concert a lieu. Ce plan est préalablement réalisé dans un logiciel de dessin ou d'édition d'images et sauvegardé dans un format GIF, PNG ou JPG.

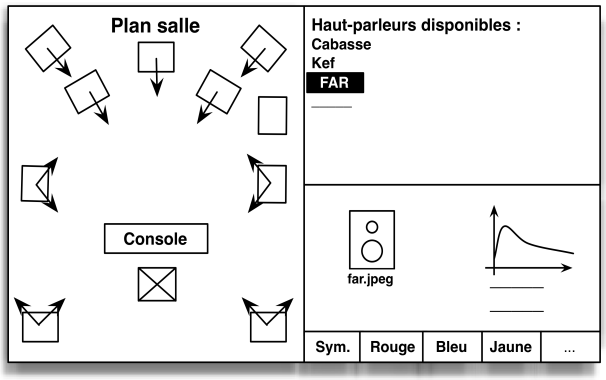


Figure 3. Interface de placement des haut-parleurs

Dans la partie droite de l'écran, on trouve l'intégralité des haut-parleurs disponibles pour la constitution de l'acousmonium. Cette liste est maintenue à jour dans un fichier XML [14] dont la structure est définie par un schéma XSD [14].

L'utilisateur peut, dès lors, sélectionner dans cette liste un haut-parleur spécifique et le positionner sur le plan. Il peut ensuite modifier son orientation dans la direction souhaitée. Pour chaque enceinte ainsi disposée sur le plan, on affiche ses caractéristiques propres (bande passante, marque, modèle, puissance, ...) ainsi que ses options d'installation. Ces options caractérisent leur emplacement en hauteur et leur directivité. Un code couleur donne une idée sur leur « altitude » (rouge s'ils sont situés au balcon, par exemple). Un autre code indique leur orientation: la flèche, le point (vers le haut) ou la croix (vers la bas).

Comme la plupart des haut-parleurs sont appairés et positionnés de manière symétrique, on automatise grâce à un seul bouton virtuel le placement de la seconde enceinte plus rapidement et plus précisément.

Une fois finalisé, le plan permet d'avoir une vue globale de l'installation à partir de laquelle, en répétition, on peut écouter la couleur de chacune des enceintes de manière à apprécier leur qualité. Quelques boutons virtuels permettent de sélectionner des sources sonores standards (1000Hz, bruit rose, …) qui sont envoyées dans les haut-parleurs sélectionnés.

3.2. Assignation des pistes audio et des sorties

Une fois les enceintes placées, on peut passer au deuxième onglet de la configuration. Le plan est alors figé et on ne peut plus, ni déplacer les objets, ni changer les couleurs. Seule la sélection reste disponible pour permettre l'affectation des pistes audio de lecture et des sorties de l'interface audio-numérique. On définit de cette façon pour chaque haut-parleur quelle piste de lecture lui est assignée et quelle sortie physique de l'interface audio-numérique lui correspond.

Comme on peut le voir dans la figure 4, l'ensemble de ces pistes et sorties apparait dans la partie droite de l'écran. Il suffit alors de sélectionner chaque piste de lecture et de la lier à son enceinte. On associe par une manipulation similaire la sortie audio physique de l'interface à laquelle le haut-parleur ou son amplificateur est relié.

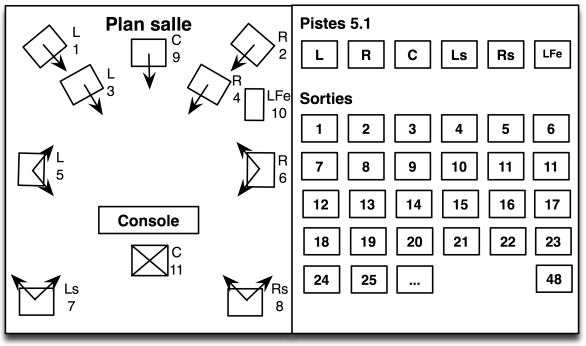


Figure 4. Interface d'assignation des pistes et sorties

Sur la partie gauche de l'écran, le plan se complète avec les informations d'assignation. Les pistes de lecture sont codifiées selon les habitudes utilisées en cinéma ou en vidéo. Par exemple, pour une oeuvre en 5.1 on utilise les lettres L, R, C, Ls, Rs, Lfe. Pour les sorties audio on utilise les numéros inscrits à l'arrière de la carte son.

On réalise cette étape avant le montage, en considérant la « playlist » prévue dans le cadre du concert. Par exemple, on envisage une configuration générique à partir d'une pièce en stéréo, une en 5.1 et une autre en octophonie. Ces assignations de pré-configuration, sauvegardées dans des fichiers de pré-réglages peuvent aisément être rappelées et éventuellement modifiées en fonction des demandes particulières de chaque interprète ou de certaines directives imposées par le compositeur.

Pendant le concert, ces configurations seront rapidement chargées entre chaque oeuvre interprétée.

3.3. Assignation des curseurs

Une fois les haut-parleurs placés et configurés, on passe dans la troisième phase correspondant au troisième onglet du programme. La partie gauche de l'écran reste inchangée et la partie droite présente les curseurs disponibles sur la table de mixage. Il suffit d'en sélectionner un pour lui assigner le haut-parleur qu'il va contrôler.

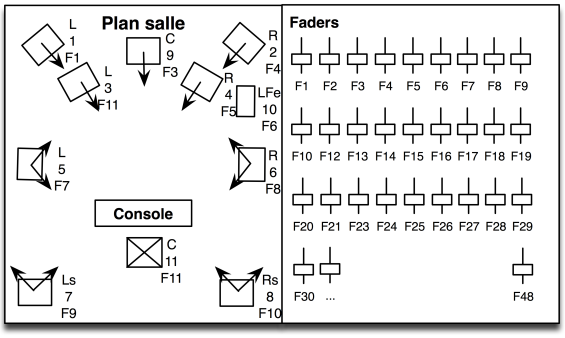


Figure 5. Interface d'assignation des curseurs

Si on dispose d'une surface de contrôle numérique on peut aisément imaginer pouvoir toucher directement le curseur physique avant de l'assigner. Mais ergonomiquement, cette technique risque d'être moins confortable puisqu'elle nécessite plus de déplacements entre écran et table de mixage.

Au terme des ces trois parties de configuration préparées avant le concert, le logiciel offre la possibilité d'imprimer les fiches techniques, les plans de câblage et de disposition des diffuseurs. Ces impressions permettent de constituer rapidement les dossiers nécessaires aux techniciens et aux régisseurs pour le montage. Elles permettent également de présenter aux interprètes les documents récapitulatifs des affectations réalisées pour la diffusion de leurs morceaux.

Avant le concert, ces configurations peuvent éventuellement être retouchées en fonction de contraintes particulières rencontrées lors de l'installation ou en fonction de désidératas de certains interprètes ou compositeurs.

3.4. Ajustement de la balance

La balance consiste à régler les niveaux de sortie de l'interface audio pour donner un équilibre en terme de volume sonore à l'ensemble des enceintes lorsque les curseurs de la table de mixage sont au maximum (0 dB). Chaque niveau de sortie des paires d'enceintes doit être réglé de façon à obtenir des changements d'espace homogènes. La manière la plus ergonomique pour cette opération reste l'utilisation de curseurs physiques. Les curseurs virtuels ne sont pas suffisamment précis et réactifs pour ce réglage très sensible. L'utilisateur, en effectuant une série de tests d'écoute à la console de mixage, affine le niveau de sortie de la carte son grâce au quatrième onglet du volet droit de l'interface. Les modifications du niveau de sortie se font sur la table

multi-touches avec un curseur virtuel proposant une large course pour augmenter la précision de ce réglage.

A cette étape on peut enregistrer plusieurs « presets » de balance qu'on rappellera à souhait pour essayer un maximum de combinaisons possibles afin de garantir le meilleur équilibre du système global.

3.5. La spatialisation

La cinquième étape ne sera pas décrite dans cet article. Elle consiste en la finalité de cette configuration: la spatialisation de l'oeuvre en concert. Nous travaillons actuellement sur ce projet pour proposer de nouvelles techniques plus intuitives et plus ergonomiques, basées sur les surfaces de contrôle multi-touches.

4. LE PROGRAMME MAX/MSP

Le « patch » réalisé dans Max/MSP gère la transformation des messages OSC qui viennent de l'interface graphique en événements MIDI à destination de l'interface audio. Cette couche logicielle pourrait être évitée dans notre cas mais nous avons préféré ne pas fermer notre application à notre seul problème. L'avantage de cette couche est qu'elle permet dans d'autres cas de figure un fonctionnement avec d'autres interfaces, voire même éventuellement d'intégrer l'ensemble lecture audio / routage / mixage dans un seul et même programme.

Le seul inconvénient actuel réside dans la latence introduite dans la chaîne, ce qui n'est pas un facteur critique au moment de la configuration.

5. RÉSULTATS ET CONCLUSIONS

Le système actuel permet de préparer la configuration d'un acousmonium avant son installation en envisageant toutes les possibilités imposées par les oeuvres diffusées lors du concert.

Il permet :

- un gain de temps important lors de la préparation d'un concert et de l'élaboration des documents techniques pour le montage
- aux techniciens et régisseurs d'avoir des documents plus précis et plus complets qui les aident lors de l'installation
- un gain de temps lors de la configuration du système pendant les répétitions
- à l'interprète d'avoir une représentation plus claire du dispositif grâce à des outils intuitifs et visuels, tout en répondant rapidement à ses demandes spécifiques en réalisant ses propres adaptations
- une ouverture vers de nouvelles perspectives au niveau du contrôle de la spatialisation, à travers de nouvelles interfaces plus ergonomiques et plus intuitives qui pourraient avantageusement compléter l'utilisation de la console de mixage

6. RÉFÉRENCES

[1] Baalman M. "Spatial composition techniques and sound spatialisation technologies", *Organised Sound* **15**(3): 207-218, Cambridge University Press, 2010.

[2] Kendall G. "Spatial Perception and cognition in multichannel audio for electroacoustic music", *Organised Sound* **15**(3): 228-238, Cambridge University Press, 2010.

[3] Wilson S. and Harrison J. "Rethinking the BEAST: Recent developments in multichannel composition ElectroAcoustic Sound Theatre", *Organised Sound* **15**(3): 239-250, Cambridge University Press, 2010.

[4] Schumacher M. and Bresson J. "Spatial Sound Synthesis in computer-Aided Composition", *Organised Sound* **15**(3): 271-289, Cambridge University Press, 2010.

[5] Ramakrishnan C. "Zirkonium: Non-invasive software for sound spatialisation", *Organised Sound* **14**(3): 268-276, Cambridge University Press, 2009.

[6] Normandeau R. "Timbre Spatialisation: The medium is the space", *Organised Sound* **14**(3): 277-285, Cambridge University Press, 2009.

[7] OSC (Open Sound Control) 2011. http://opensoundcontrol.org.

[8] Tarasti E. "L'espace dans le discours musical", Espaces, *Les cahiers de l'IRCAM*, Paris, 1994.

[9] Jullien J-P. et Warusfel O. "Technologies et perception auditive de l'espace", Espaces, *Les cahiers de l'IRCAM*, Paris, 1994.

[10] Vandegorne A. "L'espace comme cinquième paramètre musical",

[11] Max/MSP (Cycling '74), http://www.cycling74.com/, 2011.

[12] Bayle F. "La musique acousmatique ou l'art des sons projetés", Encyclopedia Universalis, 1984. Révision dans : Bayle François, Musique Acousmatique, propositions… …positions –, Paris, Editions Buchet/Chastel – INA, 1993, p. 47-68.

[13] Bosi M. "An Interactive Real-Time System for the Control of Sound Localisation", Proceedings ICMC 1990, 1990, p. 112-114.

[14] Extensible Markup Language (W3C) http://www.w3.org/XML/, 2003.

[15] Open Sound Control (OSC) http://opensoundcontrol.org/, 2011

[16] Processing http://processing.org/, 2011

[17] TUIO http://www.tuio.org/, 2011

[18] Serial Multichannel Audio Digital Interface (MADI) http://www.aes.org/, 2011

La partition mobile SVG : un premier bilan typographique et musical

Mike Solomon
mike@apollinemike.com
Conservatoire de Dunkerque et École de Musique de Saint-Chamond

ABSTRACT

Un nouveau né dans le monde musical, la partition mobile ou animée commence à battre son plein dans le domaine du spectacle vivant, soulevant ainsi des questions technologiques et esthétiques sur la composition, l'interprétation et la diffusion de ces partitions. Cet article a pour objet de fournir un aperçu de mes expériences avec ces partitions, discutant des grandes lignes de l'approche utilisée pour créer ces partitions ainsi que la réception de ces partitions auprès des musiciens et du public.

1. INTRODUCTION

La partition mobile change de manière fondamentale le rapport entre un document musical et ceux qui interagissent avec ce document. Elle va au-delà de la notation graphique en nous forçant à nous interroger sur les outils d'analyse, lecture et interprétation qui peuvent nous aider à encadrer et trier un débit d'informations musicales qui se défile à vingt-quatre images par seconde. En tant que texte artistique, elle nous force également à se demander qui a le droit de regarder cette partition pendant un spectacle vivant et par quel biais elle devrait être diffusée. Cet article répond à ces questions en résumant mes expériences avec des partitions mobiles créées à partir de GNU LilyPond et Python en utilisant la spécification SVG. Il esquisse un cadre philosophique dans lequel ces partitions s'inscrivent en discutant des technologies qui permettent de les créer et les diffuser ainsi que les questions ouvertes que soulèvent ces technologies.

2. BILAN DU BILAN – LA CHRONOLOGIE DES PARTITIONS MOBILES SVG

Les recherches pour cette article rentrent dans la catégorie de l'« action research » et se sont effectuées au cours de la dernière année dans le cadre de plusieurs concerts qui présentaient mes œuvres mobiles. J'ai composé quinze morceaux animés pour des solistes et des ensembles de chambre aux Etats-Unis, en Angleterre et en France. Un concert entier y a été consacré et elles ont été également programmées dans le cadre des concerts de musique électroacoustique et congrès d'informatique musicale. Les premières partitions (*norman (âge 1)* pour clarinette solo et *MUB atrium — 2 :21am* pour ensemble de chambre) ont utilisé des technologies web comme JavaScript et PHP pour générer et diffuser les informations musicales en temps réel dans un navigateur SVG (comme Opera, Google Chrome ou Firefox) alors que les œuvres les plus récentes (i.e. *patchy the autobot* pour trio et *six cercles* pour sextet) n'ont utilisé le standard SVG que pour le rendu des images vectorielles, optant pour un convertisseur vidéo pour les diffuser en concert et sur Internet (YouTube).

3. LA PARTITION MOBILE ET LA MISE EN PAGE MUSICALE

Les catégories de gravure musicale selon lesquelles les partitions sont souvent classées s'appliquent difficilement à la partition mobile. Comme une partition sur un support papier, elle peut se servir des règles de la gravure musicale traditionnelle tout en employant des conventions de la musique graphique afin d'achever le même résultat que l'on voit chez Cardew (Treatise) ou Applebaum (The Metaphysics of Notation). Admettons que n'importe-quelle vidéo peut-être interprétée en tant que partition, cet article se concentre sur des vidéos dont la mise en espace des glyphes et symboles est faite en s'appuyant sur le savoir-faire d'un interprète qui sait lire ces objets dans une partition « statique » et qui doit transposer ses compétences à un cadre mobile.

La partition demeure donc un objet à être lu et compris afin de reconstituer ou explorer une idée sonore. Elle doit, par conséquent, être conforme aux règles de la gravure musicale qui exigent la compréhensibilité et la lisibilité [1]. Les objets dans une partition doivent être dessinés de façon à ce que l'interprète puisse les reconnaître facilement, et ces objets doivent s'enchaîner sur l'axe horizontal afin d'éviter une lecture dont la vitesse est volatile. La même idée est évoquée par Cole [2], qui préconise que le déchiffrage musical n'est possible qu'avec des informations sur la densité des événements (qui se traduit en rythme) et l'interaction entre le corps et l'instrument (qui se traduit d'habitude en hauteur). La musique ne peut pas donc être l'objet des expériences typographiques dans un cadre de spectacle vivant. Ceci étant, elle doit entre inspirante pour l'interprète, qui en regardant la partition doit être ému par la polices de caractères choisie et sa mise en espace. Cette dimension affective des polices de caractères existe dans la gravure musicale depuis des siècles, permettant aux maisons d'édition de se distinguer et de suggérer de différents comportements expressifs.

Cette double contrainte de l'efficacité de la lecture et la beauté d'une mise en page se trouve au cœur de la partition mobile. Le cheminement des symboles musicaux doit faciliter la lecture et doit en même temps inspirer l'interprète. Il y a une troisième contrainte qui se rajoute — celle de la validité de la démarche par rapport aux partitions écrites.

Autrement dit, la question se pose toujours « quels sont les acquis d'une partition mobile par rapport à une partition fixe ». Pour faire face à la première contrainte, j'ai utilisé une technique que j'appelle la « gravure interpolée ». Cette technique utilise des données de mise en page traditionnelle pour faire des interpellations sinusoïdales entre le même objet situé à deux points différents dans le temps. Par exemple, les barres de ligature dans la gravure musicale traditionnelle sont contraintes par plusieurs règles d'espacement. Leurs pentes doivent être amorties en fonction de la convexité des hampes qu'elles relient et elles doivent toujours commencer et finir soit sur une barre de portée, soit juste en-dessous ou au-dessus d'une barre. Cette dernière règle est particulièrement difficile à enfoncer pour les triple et quadruple croches, qui ont des pentes légèrement différentes pour permettre aux barres de tomber aux bons endroits sur la portée. Dans le cadre d'une partition mobile, les barres de ligature animées doivent préserver cette relation avec la portée si elles vont déclencher la même réponse de la part de l'interprète. Pour ce faire, j'ai choisi des barres de ligature « cibles » entre lesquelles je fais des interpolations dans le temps. La même procédure est valable pour les hampes uniques dotées des croches. La gravure traditionnelle exige qu'il y ait une distance minimale entre ces objets pour faciliter la lecture et cette distance doit être respectée aussi dans les partitions animées.

Quant à la deuxième contraint — celle de l'élégance de la mise en page comme facteur esthétique dans l'interprétation musicale — la partition mobile fournit une abondance d'informations esthétiques qui peuvent finir par distraire l'interprète. Des changements dans le champs visuel ont tendance à prévaloir par rapport au champs auditif, et les interprètes peuvent facilement se laisser emporter par des éléments de la partition qui n'ont pas d'impact sur ce qu'il faut jouer. Le défi devient donc de créer un résultat visuel intéressant tout en permettant à l'interprète d'avoir les repères nécessaires pour se canaliser sur les bonnes informations dans la partition.

Les bases de la réponse à la question sur la validité générale de la partition vidéo ont déjà été partiellement établies par la notation graphique. Dans les deux formes d'expression, le compositeur cherche à engendrer des réponses musicales différentes et de créer des œuvres d'art mixtes où l'expression artistique est à la fois sonore et visuelle. La différence entre des partitions mobiles et la notation graphique se trouve dans le vocabulaire gestuel implicite dans une vidéo. Il est très difficile de projeter le sens d'un geste sur un support figé comme le papier. On a le problème inverse avec une partition mobile ; le langage visuel suggère beaucoup de gestes qui n'ont pas forcément de lien avec les gestes qui conviennent à la production sonore mais qui influent quand même sur la façon dont l'interprète répond à la partition.

4. LA PARTITION MOBILE DANS LE CADRE DU SPECTACLE VIVANT

Plusieurs problèmes associés à la partition mobile au sein du spectacle vivant ont d'ores et déjà été abordés avec d'autres médias fixes. Des morceaux pour bande et instrument, par exemple, entravent la souplesse d'interprétation dans la mesure où on ne peut ni ralentir ni accélérer l'écoulement du morceau dans le temps. Plusieurs œuvres évitent cette fixité en utilisant de l'improvisation comme outil de contrôle temporel. L'interprète doit accomplir un certain nombre de tâches dans un cadre temporel fixe, mais il y a une souplesse par rapport au temps pris pour chaque tâche et même le parcours entre les tâches. Dans mes œuvres mobiles, j'ai utilisé le même stratégie pour que l'interprète ne se sente pas contraint par les aspects fixes des morceaux. Par exemple, *patchy the autobot* utilise des barres de reprise qui renvoient le lecteur toujours vers le début de la partition. Ce qui change, c'est le contenu de la partition après chaque lecture successive. Bien que les événements arrivent toujours aux mêmes instants dans la vidéo, l'interprète peut changer la vitesse de sa lecture pour prendre plus ou moins de temps avec ce qui lui intéresse.

Il y a cependant des différences importantes entre les challenges d'une partition mobile et ceux d'un morceau pour bande. Il faut d'abord considérer la question de la diffusion des partitions pendant un concert. Les interprètes peuvent tous lire la partition à partir d'un écran géant ou bien à partir des appareils séparés (des iPads, par exemple). Les deux options changent le rapport entre l'interprète et sa partition ainsi que la mise en scène de l'événement. Pour les œuvres interprétées à partir d'un écran, l'interprète ne peut pas avoir ses propres indications musicales écrites sur la partition et doit alterner son attention entre l'écran et ses collègues. Il ne peut pas non plus s'orienter à cent pourcent vers les spectateurs car il doit toujours pouvoir voir l'écran qui doit être écarté pour que les spectateurs puissent le voir. Avec un support comme l'iPad, l'interprète peut avoir un pupitre numérique devant lui, mais on rentre dans une sorte de voyeurisme informatique. Quand on regarde quelqu'un en train de lire la musique sur un support papier, on n'est pas forcément intéressé par ce qui est écrit sur la partition. En revanche, quand on regarde quelqu'un en train de regarder un support numérique, le spectateur se sent inéluctablement exclu d'un circuit de communication entre l'écran et l'interprète.

La question se pose aussi du rôle de la partition vis-à-vis les spectateurs. L'on ne voit que rarement des concerts où une partition fixe est affichée ou distribuée aux spectateurs pour qu'ils puissent la suivre en temps réel. Avec chaque concert que j'ai fait de la musique mobile, en revanche, les organisateurs veulent toujours que la salle puisse voir la partition. Cela entraîne un certain nombre de phénomènes inhabituels au sein du spectacle vivant. Dans un premier temps, toutes les fautes que l'interprète est susceptible de faire sont visibles aux spectateurs. La prise de conscience de l'interprète par rapport à cette ouverture musicale a un impact sur la façon dont il joue. Les spectateurs n'ont pas non plus le même rapport avec la musique puisque le support visuel attire forcément une partie de leur attention qui aurait sinon être consacrée à l'écoute. Il y a aussi un décalage entre les informations visuelles qu'ils reçoivent et le vrai sens de ces informations pour les interprètes. Les partitions ne s'affichent pas avec leurs modes d'emploi et les spectateurs ne savent pas donc le rapport entre les dé-

placements dans la partition et les gestes des interprètes. Lorsqu'ils écoutent et ils regardent, ils inventent un rapport entre ce qui est joué et ce qui est affiché. La partition mobile devient donc un art où les outils de production devient une partie du spectacle vivant et le ressenti dépend autant de la musique jouée et l'image projetée que de la relation évoluante entre le spectateur et le système de notation qu'il découvre en temps réel.

5. NOVELLES VOIES DE RECHERCHE

Il est difficile de dégager de tendances importantes de la partition mobile vu le nombre faible de compositeurs travaillant dans le domaine. D'après les expériences énumérées ci-dessus, il y a trois axes de recherche qui semblent prometteurs :

- Des recherches sur l'attention du lecteur par rapport à la vitesse et la densité des informations visuelles.
- Une enquête sur le rapport entre les spectateurs et le morceau lorsque la partition est affichée sur un écran.
- Un ensemble d'outils qui fournit un certain nombre de raccourcis pour créer des partitions visuelles tout en gardant la souplesse qu'a un logiciel comme Max/MSP ou OpenMusic.

La plate-forme informatique qui m'a permis de créer ces morceaux est fait dans l'optique d'accomplir ce dernier but. Écrit en Python, il permet de créer des « gravures interpolées » à partir des partitions LilyPond simples et de transformer l'image vectorielle qui en sort en plusieurs formats vidéo. Il faudrait toutefois que cette approche soit généralisée autant que possible pour permettre aux compositeurs de faire des expériences avec les technologies et explorer d'avantage les possibilités de cette nouvelle voie de mise en page et de création musicale.

6. REFERENCES

[1] E. Grazi, *Doctoral Dissertation*. Rome : Unpublished, 1912.

[2] H. Cole, *Sounds and Signs*. London : Oxford University Press, 1974.

LE GLISSEMENT PLASTIQUE DE L'ENJEU D'INTERPRÉTATION DANS LES ARTS DES SONS : POUR UN FORMAT D'INDEXATION VERTICALE

Romain BRICOUT
EDESAC – CEAC
Université de Lille
romain.bricout@univ-lille3.fr

RÉSUMÉ

Articulée en deux mouvements, cette contribution propose dans un premier temps une analyse musicologique des pratiques électroacoustiques considérées dans leur plus grande diversité. L'influence de l'enregistrement sonore, originellement « dissociatif », y tient en effet un rôle essentiel dans l'apparition de certaines esthétiques musicales et sonores lorsqu'il est couplé à des pratiques de création ou de réception spécifiques. Au sein de notre analyse, les effets de la phonofixation sont cependant disctincts de la plasticité du signal sonore induite par sa transformation en signal électrique. Basée sur les observations d'une étude systématique de l'organologie électroacoustique, nous proposons une classification des outils de création sonore spécifiquement plastiques en trois grandes catégories : « sources sonores », outils de « transformations » et enfin de « fusions » sonores. Ouvrant la voie à une forme de « glissement plastique », le déplacement de l'activité artistique au sein de ces catégories occupe une place fondamentale dans le compréhension de l'évolution profonde des esthétiques musicales au XX^{e} siècle. Fort de ces constats théoriques, c'est dans un second temps que nous suggérons certains aspects de développement technologique que pourront intégrer les supports phonographiques de demain dans l'idée d'un format d' « indexation verticale ».

1. PRATIQUES DU PHONOGRAMME

« Reste que le phonogramme est d'abord et massivement l'une des conditions de possibilité de l'apparition de ce que j'appelle la misère symbolique : une situation sociale tramée par ces hypomnémata machiniques, et qui est en cela, c'est-à-dire comme issue de ce tournant machinique, caractérisée par le fait d'une perte de participation esthétique, celle-ci étant elle-même induite par le processus de perte d'individuation dont Simondon a formé le concept en analysant la situation du prolétaire : cette perte d'individuation résulte de la transformation, par les machines et les appareils, du monde du travail, à partir du XIX^{e} siècle, et, aujourd'hui, du monde de tous les jours, en tant qu'il est devenu le monde de la consommation et du tournant machinique de la sensibilité. »[1]

Avec l'apparition des technologies de phonofixation, il n'est plus nécessaire de savoir jouer de la musique pour en écouter. Là où la partition nécessitait en effet toujours une forme d'actualisation par le jeu ou la lecture[2] de l'extériorisation de la mémoire[3] musicale qu'elle représente, le support phonographique externalise cette fonction d'actualisation au sein d'une machine, constituant en cela un véritable « tournant machinique de la sensibilité »[4]. Cependant, même si le phonogramme s'avère en effet essentiellement dissociatif (séparant les producteurs/créateurs de musique d'un côté et consommateurs/récepteurs de l'autre), il nous paraît ici important de rappeler qu'il est aussi, dans le même temps, la condition de possibilité de l'apparition de certaines musiques *inouïes*. Le phonogramme reste par exemple ce *pharmakon*[5] qui permet à la fois l'apparition de la « misère symbolique » et de la musique concrète. Soigné par cette forme de pratique qu'est l'écoute réduite [6], le *pharmakon* qu'est le phonogramme engendre la musique concrète. Les pratiques pharmacologiques apportées à la phonofixation ne s'arrêtent cependant pas à ce seul exemple : la phonofixation est aussi la condition d'apparition de l'ethnomusicologie moderne, utilisant l'enregistrement pour étudier des traditions musicales orales dont la transcription écrite serait peu

[1] Stiegler B. *De la misère symbolique, 2. La catastrophè du sensible*, p. 50 [6]

[2] Il existait aussi l'autre possibilité de réception que représente la situation de concert, mais celle-ci restait d'une part très rare en comparaison des pratiques d'écoutes actuelles des musiques fixées sur supports et, d'autre part, cette écoute de concert n'était passive qu'en apparence : l'apprentissage de la musique étant alors *institué*, le dispositif du concert était articulé à des pratiques préalables de déchiffrage et de jeu éventuel de transcriptions ou réductions des partitions orchestrales.

[3] Comme « hypo-mnèse » réalisée sur des supports de mémoire.

[4] Cf. Donin N. & Stiegler B., « Le tournant machinique de la sensibilité musicale » [4].

[5] Terme grec désignant le caractère bivalent du remède, qui peut selon la *posologie*, se transformer en poison.

[6] Visant à focaliser l'attention sur les caractéristiques spectro-morphologiques (plastiques et formelles) du son, en s'émancipant notamment d'une orientation vers les causes.

rigoureuse d'un point de vue méthodologique, ou tout simplement impossible[7].

Le phonogramme permet aussi l'apparition du *BeBop*, ce qui reste flagrant chez Charlie Parker qui utilisa le phonographe comme un moyen supplémentaire dans l'apprentissage du saxophone et de l'improvisation[8] pour analyser et s'approprier les *chorus* d'autres musiciens de jazz. Rappelons ici aussi les nouvelles pratiques « d'interprétation des enregistrements » (cf. sur ce point Tiffon V. « L'interprétation des enregistrements et l'enregistrement des interprétations : approche médiologique » [7]) que développent les formes évoluées de *sampling*[9], tout comme certaines pratiques de *turntablism*, où, loin des pratiques du « disc-jockey » traditionnel, la platine et le disque sont envisagés en tant qu'instruments de musique à part entière. On trie ces disques, on les *marque* en repérant les passages intéressants, etc. pour aboutir aux manipulations concrètes du *scratch* et autres pratiques de *juggling* consistant à « jongler » littéralement avec deux enregistrements identiques pour en effectuer un re-montage en *temps-réel*[10]. L'ensemble de ces exemples illustre ici l'invention de diverses *pratiques* visant la réappropriation des technologies de phonofixation originellement dissociatives.

2. LE GLISSEMENT PLASTIQUE DE L'ENJEU D'INTERPRÉTATION DANS LES ARTS DES SONS

2.1 Phonofixation et Transduction sonore

Au travers de l'exemple particulier des expériences fondatrices de la musique concrète que sont celles de la « cloche coupée » et du « sillon fermé »[11], nous pouvons observer l'influence déterminante de la phonofixation sur la pensée musicale du XX^e^ siècle : la musique concrète ne peut en effet voir le jour *qu'avec* la phonofixation, dont les multiples sensibilités extra-musicales[12], savantes[13] et populaires[14] sont elles-mêmes issues dans l'ensemble des « arts des sons fixés sur supports ».

Apportons cependant ici quelques précisions face à ce constat : la multiplicité des opérations de création sonore purement plastiques nécessite en effet de nuancer l'influence de l'enregistrement. De nombreuses opérations de création sonore (et dont en tout premier lieu, de *transformations* sur lesquelles nous reviendrons par la suite) n'apparaissent en effet pas directement liées à la technique de l'enregistrement. C'est d'ailleurs la question que soulèvent les toutes premières formes de supports analogiques : les cylindres et disques originels des paléophones (Cros), phonographes (Edison) et gramophones (Berliner) sont des supports spécifiques de la phonofixation, mais ils ne pouvaient cependant pas, dans leur état initial, donner lieu à une véritable réappropriation (ou détournement) du support pour des activités de création. Il aurait pour cela fallut travailler à la manière de Lucius, *gravant* lui-même à l'aide d'un poinçon le sillon phonographique dans le but de recréer la voix perdue de sa jeune fille alors tragiquement disparue[15].

Il faudra ainsi attendre l'invention de l'amplification (invention de la lampe triode en 1904 par De Forest) et le développement des technologies afférentes de la *transduction*[16] du son en signal électrique pour rendre réellement possible l'apparition du « son vitesse lumière »[17]. L'analogie des variations de pression

[7] Ce qui fait aussi de cet organe technique de la phonofixation qu'est le magnétophone analogique portable (dont le célèbre Nagra) un outil transversal aux pratiques d'investigations ethnomusicologiques et aux dispositifs d'enregistrements de pièces phonographiques et d'art radiophonique (comme genres à part entière et précurseurs des arts des sons fixés sur supports).

[8] C'est d'ailleurs ici sans doute un phénomène de compensation chez Parker qui se concentra sur le disque après avoir été traumatisé par son expérience malheureuse au Reno Club, où le batteur Joe Jones jeta une cymbale aux pieds du jeune Parker lors d'une *jam-session*, lui priant de bien vouloir quitter la scène et laisser la place. A la suite de cette humiliation, Parker compense le circuit d'apprentissage de la *jam-session* par une pratique intensive et systématique du phonographe qui lui permettront d'inventer une approche révolutionnaire du jazz et du saxophone.

[9] Cf. Bricout R. « Les incarnations du *sampler* au XX^e^ siècle : l'avènement du musicien-luthier » [3]

[10] Cf. à ce sujet le documentaire *Scratch* réalisé en 2001 par Doug Pray, et plus particulièrement les interventions du *DJ* d'origine philippine Qbert.

[11] Ces expériences de « phénoménologie pratique » montrent à la fois comment la perception fonctionne en termes de rapports (le son de cloche amputé de son attaque n'est plus reconnaissable, ce qui illustre parfaitement ici le concept de « rétention primaire »), et comment elle est influencée par la mémoire (la répétition identique que suppose la mise en boucle d'un son engendre des phénomènes perceptifs différents, ouvrant ainsi la voie aux possibilités de décontextualisation et de réduction, illustrant ici l'action des rétentions secondaires sur le mode de « sélection » des rétentions primaires).

[12] La phonographie, le cinéma pour l'oreille, le *design* sonore, etc.

[13] On pourrait ici objecter que l'apparition de la musique électronique savante (Cologne, WDR), dans son utilisation de générateurs électroniques, se passe très bien de la phonofixation. Ceci reste en un sens particulièrement juste et constitue la *double filiation* des musiques électroacoustiques, leur caractère originellement « bicéphale ». Cependant, on remarquera dans le même temps que la rareté des générateurs électroniques obligeait, dans un véritable couplage des prothèses techniques, une utilisation conjointe des technologies de la phonofixation. Les musiciens ne disposaient le plus souvent à cette époque que de quatre générateurs, et les technologies de phonofixation permettaient alors de complexifier les spectres par réenregistrements successifs, jusqu'à la limite d'apparition de bruit analogique parasite (dégradation de transfert). Cf. les premiers travaux de Stockhausen, dont *Studie II* (1954), ou encore *Glockenspiel* (1953) d'Herbert Eimert.

[14] Dans la capacité éminemment efficace du disque à diffuser et confronter des esthétiques musicales qui n'avaient jusqu'alors pas la possibilité de se rencontrer.

[15] Ce personnage, rongé par le folie et le chagrin, fut inventé par Raymond Roussel dans son livre *Locus Solus* [5] en 1914 : l'auteur y témoigne par cette histoire d'une intuition fulgurante concernant l'apparition prochaine des arts des sons fixés sur supports.

[16] En biologie, la transduction renvoie à la capacité d'une cellule à convertir la nature d'un signal (ex : conversion d'un signal nerveux en signal chimique, etc.). Dans le domaine du son, microphones et haut-parleurs sont eux-aussi appelés des *transducteurs*, car ils permettent de convertir les signaux acoustiques en signaux électriques, et inversement.

[17] Pour reprendre une expression poétique de François Bayle qui caractérise l'accélération de la vitesse du son lors de son électrification. *Son Vitesse-Lumière* (1983) est d'ailleurs aussi le titre d'une oeuvre du

acoustique et des variations du signal électrique reste en effet à la base de la plasticité sonore caractéristique du travail concret, et ce, au même titre que la phonofixation dont les effets restent sans doute, selon cet angle, par trop survalorisés. Aussi, pour plus de rigueur, il nous paraît important de réaffirmer que la dénomination d' « arts des sons fixés sur supports » se doit toujours d'être accompagnée de l'adjectif « *électroniques* ». Cette expression se veut en effet généralisante : il faut en effet garder à l'esprit que le « support électronique » représente à la fois le *support de mémoire* (support physique d'inscription : disque et bande magnétique analogiques, supports numériques) et le *support de pensée* et de *pratiques*[18] qu'incarne la modulation du signal électrique. Montage (incluant les possibilités de répétition et de réversibilité temporelle) et plasticité développent bien des effets distincts, mais on attribue le plus souvent la cause de tous ces effets à la seule phonofixation : phonofixation et transduction sonore constituent un couplage technique indissociable dont la double nature ne doit pas être négligée[19].

2.2 Sources – Transformations – Fusions

A la suite d'une étude systématique des outils de l'organologie électroacoustique, nous avons proposé[20] une classification en trois catégories des opérations portant sur un travail sonore purement plastique. Revenons rapidement sur ces dernières. La catégorie des « sources sonores » peut dans un premier temps relever d'une approche plutôt concrète (nécessitant un dispositif du type corps sonore - microphone(s) – enregistreur) ou électronique (regroupant l'ensemble des générateurs et autres oscillateurs électroniques). La deuxième catégorie des « transformateurs » se divise en plusieurs sous-ensembles incluant les traitements spectraux (filtrages, etc.), traitements temporels[21] (réverbération, effet de chœur, *phasing*, etc.) et enfin traitements dynamiques (distorsions, compression, normalisation, etc.). La dernière catégorie des « fusions » regroupe enfin en son sein les différentes pratiques de mixage (fusion *électrique* des signaux sonores) et de spatialisation (fusion *acoustique* des différents sons). Ces différentes opérations *composent* ainsi le processus de création sonore plastique. Notons qu'au cours de ce processus de création, et par l'utilisation conjointe de l'enregistrement, le produit de chaque étape peut constituer la source de l'étape précédente (le produit d'un mixage de plusieurs sons pouvant être appréhendé comme une nouvelle source sonore à transformer, et ainsi de suite) réalisant de véritables *surimpressions* sonores et gestuelles. La « spécialisation » des outils inhérente à cette division des tâches introduit en effet des changements majeurs quant à l'appréhension de la poïétique musicale électroacoustique. L'« instrument » traditionnel se retrouve littéralement « explosé » en un foisonnant réseau de machines au sein du « studio » -réel ou virtuel- de création[22]. Il s'agit ici pour nous de comprendre en quoi ces modifications organologiques fondamentales peuvent constituer de nouveaux supports de *pensée* et de *pratiques*, et quelles sont les conséquences de ces mutations sur le geste musical lui-même.

compositeur.

[18] Cf. Bricout R. *Les enjeux de la lutherie électronique : de l'influence des outils musicaux sur la création et la réception des musiques électroacoustiques*, Thèse de doctorat, Ecole Doctorale SHS, Université de Lille, 2009 [2].

[19]Au delà des esthétiques concrète et électronique, c'est d'ailleurs peut-être dans cette double nature que s'origine le caractère « bicéphale » précédemment évoqué de la musique électroacoustique.

[20]Cf. Bricout R. *Les enjeux de la lutherie électronique, op. cit.* [2].

[21]Il convient ici de préciser que cette catégorisation est organisée selon la cible des actions effectuées. Dans le cadre de ces transformations essentiellement plastiques, ces transformations « temporelles » ont plus d'incidence sur la matière que sur le temps.

[22]Rappelons toutefois que Pierre Schaeffer, dans sa tentative d'instauration de nouvelles pratiques *associées* de la phonofixation et de l'organologie électroacoustique, définira l'oreille comme instrument principal du musicien concret, et ce bien avant le microphone, le magnétophone, ou même le studio tout entier.

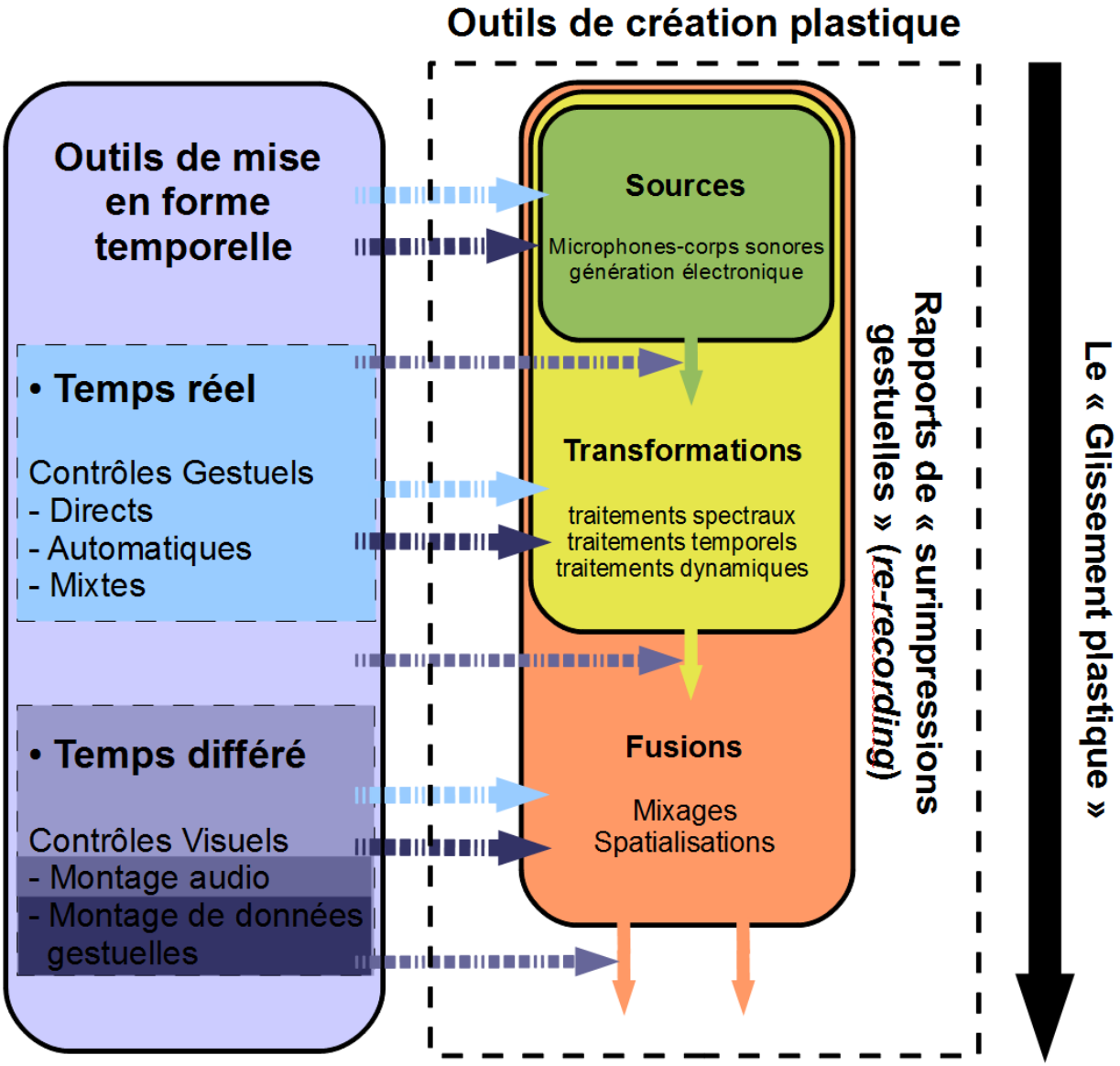


Figure 1. Organologie de la création électroacoustique : le "Glissement Plastique" - schéma récapitulatif.

2.3 Premiers « glissements »

De manière corrélative à l'apparition des différents outils, un *mouvement* de fond de l'évolution des pratiques musicales nous est apparu des sources sonores aux opérations de fusions, en passant par celles de transformations. Dans le véritable déplacement de l'enjeu d'interprétation musicale qu'il constitue, nous appellerons ici ce mouvement « glissement plastique » en référence au processus qui le caractérise. Là où la source sonore représentait en effet l'enjeu d'interprétation principal[23] de la musique instrumentale « traditionnelle », les opérations de transformations ont ensuite déterminé l'esthétique du travail concret comme première véritable « révolution » du sonore. Les opérations de « fusions » sonores incarnent enfin la « pointe » de ce glissement plastique : elles correspondent actuellement aux principales formes d'interprétations électroacoustiques, en l'occurrence des pratiques de spatialisation sur acousmonium et autres dispositifs de haut-parleurs. Tel est le mouvement du glissement plastique, des pratiques de la musique traditionnelle à celles de la musique concrète pour aller vers celles de la musique acousmatique.

Selon cette analyse, et dans sa mise en perspective avec une étude plus précise du développement des pratiques musicales scéniques (en tant que dispositifs de diffusion), on peut observer que la musique concrète n'a pas utilisé la forme du concert[24] en tant que véritable médium de son esthétique. User du concert pour en faire un vecteur de l'esthétique concrète aurait en effet signifié de « mettre en scène » les *pratiques* concrètes. Or, dès leurs débuts, les concerts de musique concrète donnés par Pierre Schaeffer et Pierre Henry ont proposé des pratiques de spatialisation du son (fusions sonores), notamment *via* l'usage du « potentiomètre d'espace »[25], sans toutefois véritablement développer de nouvelles formes d'interprétation des pratiques de transformations sonores en public. Ceci peut dans un premier temps s'expliquer par l'aspect rudimentaire des outils de transformations de l'époque : les pratiques d'interprétation mettant en œuvre de telles opérations de transformation n'auraient sans doute, dans une configuration de temps réel, été que trop pauvres[26]. Revenons aussi sur l'influence d'une certaine « fétichisation » de la phonofixation qui permet un *contrôle* (presque) total de la diffusion des oeuvres : le compositeur n'a plus à passer par le prisme plus ou moins déformant de l'interprète pour transmettre sa pensée musicale. Le médium du concert véhiculant *autre chose* que l'esthétique concrète elle-même, le processus de transmission de la pensée concrète, comme investissement des opérations de transformations sonores, nous paraît ici incomplet tout en anticipant sur l'avenir : le « concert concret » est, avant tout, et dès le départ, déjà un concert acousmatique.

Comme première « révolution » du glissement plastique, la réappropriation des opérations de transformations sonores au sein des pratiques scéniques semble cependant avoir été prise en charge quelques années plus tard par le champ des musiques populaires : il y avait en effet là, dans le processus de développement historique de la musique, un créneau vacant que la musique savante n'avait pas occupé. Dès la fin des années 60, l'électrification de l'instrumentarium populaire engendre en effet des mutations stylistiques importantes, allant du folk de Bob Dylan au jazz de Miles Davis. Jimi Hendrix, véritable « héraut » de cette époque, virtuose de la guitare électrique et des *transformations* sonores qu'il lui avait alors associées, suscita à ce propos un réel phénomène de fascination. L'électrification représentait premièrement une mutation sans précédent de la puissance sonore instrumentale[27] : c'est l'ère du « power trio »[28] où trois musiciens suffisent désormais à la constitution d'un groupe, ainsi que celle des premiers « méga-festivals » de plein air permis par le développement des capacités et des pratiques musicales d'amplification. Cette surenchère de puissance sonore provoquée par l'amplification reste à certains égards assez paradoxale[29], même si elle s'avère, elle aussi,

[23]Nuançons ici notre propos : il existe bien des pratiques de fusions sonores « traditionnelles » dont le chef d'orchestre s'avère le principal *instrumentiste* (et ce même si la figure du chef reste relativement récente à l'échelle de l'histoire de la musique).

[24]Remarquons aussi, d'un point de vue plus médiologique, que l'utilisation de la forme concert constitue un véritable « effet diligence » pour la musique concrète. Le « concert » électroacoustique utilise l'efficacité d'un dispositif spécifique de la graphosphère pour palier au développement encore trop peu avancé des dispositifs de l'audiovidéosphère. Dans un développement médiasphérique cohérent, les musiques du son (elles mêmes originellement issues de l'art radiophonique) se doivent cependant d'investir les dispositifs de diffusion audiovidéosphériques représentés par la radio-diffusion : ceci passe aujourd'hui par un développement de la diffusion multicanal, accusant malheureusement un certain retard en France.

[25]Le « potentiomètre d'espace », conçu par Pierre Schaeffer et Jacques Poullin pour les premiers concerts de musique concrète (1951), témoignait d'un fonctionnement assez proche de celui de la captation des variations électromagnétiques utilisée par le Thereminvox. Par une action sur les panoramiques, il permettait à un interprète (Pierre Henry en l'occurrence) de « projeter » le son en trois dimensions, selon des mouvements analogues à ceux de la main entre des cerceaux matérialisant l'espace de projection.

[26]Quoique correspondant beaucoup plus à une esthétique électroacoustique que concrète (les américains n'avaient en effet pas perçu la problématique de l'opposition « conceptuelle » qui opposait les esthétiques concrète et électronique), c'est en un sens ce que l'on peut déceler dans les premières oeuvres de la *Tape Music* se limitant la plupart du temps à l'exploitation d'un seul « effet » ou d'une seule transformation par pièce. Ainsi des oeuvres *Low Speed* (1952) d'Otto Luening, ou encore *Sonic Contours* (1952) de Vladimir Ussachevsky, essentiellement basées sur des effets de vitesse de lecture et de *delay*.

[27]La guitare, sous sa forme électrique, prit en effet une certaine revanche sur l'histoire : au début du XX^{e} siècle, la guitare « espagnole » était pratiquement bannie des orchestres en raison de sa faible puissance sonore. Comparé à celui des cuivres, le très faible volume de la guitare la condamnait à des phénomènes de masquages quasi-permanents. Passé le cap de son électrification, cette tendance s'inversa pour faire de la guitare l'un des instruments les plus populaires de la seconde partie du XX^{e} siècle.

[28]Citons ici justement The Jimi Hendrix Experience, ou encore le groupe Cream (Ginger Baker, Jack Bruce, Eric Clapton), etc.

[29]Il nous a en effet toujours paru étrange que les outils d'amplification, permettant un contrôle de la dynamique sonore des plus fins (ce que les pratiques savantes ont pour leur part très bien exploité), ne soit le plus souvent employés qu'à l'instauration d'une débauche de décibels

indissociable de la plasticité du signal sonore électrique. Bien plus que le gain de puissance acoustique introduit par l'amplification, le développement des pratiques d'interprétations liées aux opérations de transformations sonores nous semble en effet dans un second temps constituer le véritable objet de fascination dont l'électrification instrumentale a pu faire l'objet. Jimi Hendrix fut l'un des premiers guitaristes à *coupler* son jeu de guitare à un *jeu* des transformations sonores, développant sa virtuosité à travers une maîtrise instrumentale hors pair, mais aussi par l'invention d'un nouveau langage : celui des effets sonores. Présentés sous formes de pédales, ces effets en étaient alors à leurs balbutiements. Hendrix développera en tout premier lieu une pratique de la pédale « *wah-wah* » (filtre dont la fréquence de coupure est commandée par un mouvement du pied sur la pédale), puis lui adjoindra d'autres effets (dont le *fuzz,* forme particulière de distorsion, ou l'*octaver*, permettant un épaississement du son par transposition d'octave), allant même jusqu'à provoquer leur développement (c'est ici l'exemple de la pédale *univibe*, simulation électronique de la cabine Leslie, haut-parleur tournant originellement développé pour les orgues Hammond). Recréant de cette manière un *réseau* d'outils de transformations sonores, Hendrix fut l'un des premiers musiciens à appliquer ces mêmes opérations de transformation sur scène : couplé à une pratique instrumentale traditionnelle (la guitare en tant que « source sonore »), le jeu de Hendrix s'inscrit dans un chevauchement de « systèmes » pour le moins efficace. Dans les possibilités de filtrage inédites qu'elle introduisait au moment de son apparition, la *wah-wah* fut aussi utilisée par Frank Zappa qui avoua pour sa part l'avoir expérimentée sur pratiquement tous les instruments, sur scène comme lors de sessions de travail en studio. *Incarnée* par la pédale wah-wah, les musiciens du groupe allemand Can chanteront aussi les « louanges » de la révolution populaire du « glissement plastique » sur le morceau *Halleluwah* (1971). Dave Jackson, le saxophoniste du groupe de rock progressif Van Der Graaf Generator, utilisera un microphone pour adjoindre une *wah-wah* et d'autres effets à son jeu de saxophone, ce qu'avait aussi et dans le même but réalisé Miles Davis quelques années plus tôt sur sa trompette.

Si les musiques électroniques populaires ont pu se révéler comme autant de véritables « musiques concrètes populaires » (cf. [3]) par leurs utilisations du *sampling*, gageons que le phénomène de « glissement plastique » reste transversal à l'ensemble du xx^e^ siècle musical. L'histoire du jazz rend d'ailleurs particulièrement sensible ce passage de la pratique des sources sonores à une pratique des transformations : de l'originel *Livery Stable Blues* et son imitation des sons animaliers, au jazz-rock et au jazz-fusion des années 70, en passant par le style « jungle » de Duke Ellington et ses sonorités « exotiques »[30], jusqu'au free-jazz et ses ramifications formées par les musiques improvisées contemporaines, l'évolution du jazz épouse littéralement cette première phase du « glissement plastique ». La généalogie musicale du xx^e^ siècle reste à ce propos le plus souvent limitée à une lecture des influences stylistiques : le phénomène de glissement plastique pourrait ici proposer une grille de lecture particulièrement intéressante pour expliquer les multiples filiations traversant les musiques savantes et populaires du xx^e^ siècle (ainsi de Zappa se référant à Varèse, du lien unissant la musique de Can[31] à l'enseignement de Stockhausen, ou encore des expérimentations du rock progressif à l'origine de certains courants des musiques électroniques populaires, etc.).

Plus qu'un outil permettant d'analyser l'évolution des sensibilités musicales au xx^e^ siècle, le développement du phénomène de « glissement plastique » nous pousse à anticiper la suite du processus et à en tirer certaines conséquences sur le plan organologique. La multiplicité des exemples précédemment cités laisse en effet entrevoir l'importance que peut recouvrir un second déplacement des pratiques de transformations aux pratiques de « fusions » sonores. Le glissement plastique pose ainsi de nos jours la question de l'efficacité des dispositifs permettant la diffusion des pratiques de mixage auprès d'un plus large public et l'intégration de l'*espace* en tant que « nouveau » paramètre musical. A la suite de notre travail d'analyse, abordons ici pour terminer une proposition de développement technologique qui serait à même de venir supporter ces nouvelles pratiques.

3. POUR UN NOUVEAU STANDARD : L'IDÉE D'UN FORMAT D' « INDEXATION VERTICALE »

Dans l'étape essentielle qu'elle représente pour l'instauration d'une réelle pratique des enregistrements, l'« indexation » des supports de fixation sonore semble constituer une voie de sortie possible au caractère originellement dissociatif du phonogramme. Le projet *Semantic HIFI* mené à l'IRCAM[32] entre 2003 et 2006 tente de proposer une application concrète de cette analyse, notamment dans le développement d'une technologie capable d'identifier et de classifier les éléments structurels d'un contenu musical. Cette technologie d'indexation « horizontale », dont le but

au sein des musiques populaires.

[30]Référons nous ici à la pratique des *tricks* développée par les musiciens de Duke Ellington (dont le tromboniste Joe « Tricky Sam » Nanton) leur permettant de détourner les sonorités initiales de leurs instruments par l'utilisation de sourdines et de « bols », dont la technique d'ouverture et de fermeture devant le pavillon des cuivres portait déjà, de part son effet sonore caractéristique, le nom de *wah-wah*.

[31]Essentiellement électroacoustique, le travail de composition du groupe consiste à appréhender les prises instrumentales comme un matériau de base (source) sujet à diverses manipulations sonores et montages (ce que faisait aussi dans une certaine mesure Teo Macero pour Miles Davis...).

[32]Projet coordonné par Hugues Vinet et associant notamment Sony et Native Instruments en tant que partenaires industriels. http://shf.ircam.fr/

principal est de venir faciliter l'appropriation des supports par une écoute *active* de l'auditeur, permet d'*organiser* les pratiques de recherche et de navigation (*marquage* des couplets, refrains et ponts, en prenant l'exemple de la structure traditionnelle d'une chanson) au sein des multiples flux musicaux qui constituent la réalité de notre environnement sonore actuel. A la manière des pratiques de *turntablism*, tout un chacun se trouverait ainsi dans la possibilité de venir re-monter les morceaux, pour en proposer des versions personnelles, mais aussi de monter entre-eux des éléments issus de différents morceaux[33], ou encore de partager des données de marquage avec d'autres auditeurs, etc.

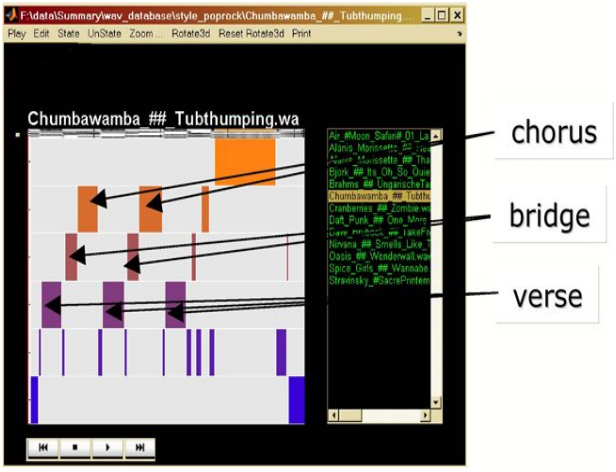


Figure 2. Exemple d'indexation horizontale (source IRCAM - Projet *Semantic Hifi*)

Même si elle représente une avancée notable quant à la réintroduction de véritables pratiques de la phonofixation pour le grand public, la segmentation induite par le format d'indexation horizontale ne nous semble cependant pas incarner une réelle prothèse associative de la phonofixation. Une segmentation grossière permet bien d'établir une analyse des formes musicales mais, par la simplification[34] des pratiques de montage que cette dernière engendre en retour, elle ne semble générer qu'une illusion d'associativité[35]. Un compositeur peut bien souvent utiliser un banc de montage pour créer sa musique, mais il ne le fait que très rarement à partir d'éléments sonores qu'il n'a (sauf exercice de style) pas créés lui-même.

Aussi, au regard de notre précédente analyse organologique montrant l'imbrication des étapes de la création plastique (sources, transformations et fusions), l'instauration d'une forme d'indexation « verticale » semblerait ici représenter une solution alternative intéressante dans la pratique des supports d'enregistrement. Appréhendée de cette manière, et ne concernant que l'étage des opérations de fusions sonores, l'indexation « verticale » serait en effet le moyen de contourner l'impossibilité de démixer[36] des réductions sonores (stéréophoniques) pour en remixer dynamiquement et/ou spatialement[37] leur constituants de manière différente. Les pratiques de « fusions » sonores (dont le mixage et la spatialisation) doivent en effet pour se développer au delà du cercle restreint des initiés être *suscitées* par les supports d'enregistrement eux-mêmes. Véritable *support* d'une pratique associative du phonogramme, un format d'indexation verticale nécessiterait toutefois le concours des créateurs/producteurs de musique pour être mis en œuvre efficacement : l'idée est ici de donner un accès aux pistes séparées d'une œuvre à même son support d'écoute, pistes sonores auxquelles seraient adjointes d'autres pistes de données informant des opérations de mixage et de spatialisation (volumes et panoramiques)[38]. L'œuvre pourrait ainsi être jouée *via* un système ou logiciel de diffusion dédié appliquant en temps réel les différentes données de mixage et de spatialisation (grâce à un algorithme spécifique, un peu à la manière de la « décompression » de certains formats[39]), et ce, suivant

[33]Pratique qui n'est cependant pas sans rappeler les formes les plus dissociatives du « zapping », la plupart du temps pourvoyeuses et/ou symptômes de misère symbolique.

[34]Les pratiques de sampling « savantes » et le *turntablism* usant en effet du même principe de segmentation, mais de manière beaucoup plus fine et évoluée. Ceci rend par exemple possible l'apparition et la pratique de la décontextualisation, comme base de l'écoute réduite.

[35]Où créateurs et récepteurs ont accès aux mêmes outils et parlent un même langage. On risque en effet ici d'encourager certaines pratiques de *sampling* « grossières » et de *mash-up.* Il est dans cette visée intéressant de se référer au livre *The Manual (How to have a number one the easy way)*, écrit en 1988 par les agitateurs d'inspiration situationniste Jimmy Cauty et Bill Drummond, formant The KLF (dont l'une des significations serait « *Kopyright Liberation Front* »). Le groupe venait alors d'atteindre le sommet des classements britanniques avec le single *Doctorin' the tardis* (sous le nom de Timelords) quand Cauty et Drummond décidèrent de dévoiler les secrets de fabrication de l'industrie musicale de masse en prenant pour exemple la réalisation de leur propre succès. Ils y expliquent comment une personne dénuée de toute compétence musicale peut obtenir un numéro 1 des classements en utilisant des samples assez grossiers de morceaux présents sur les compilations de succès des vingt dernières années et en se référant au « *Guinness Book of British Hit Singles* ». *Doctorin' the Tardis* était uniquement composé de trois samples non modifiés : l'un en provenance du *Blockbuster* (1973) de Sweet, l'autre du *Rock'n'Roll* (1972) de Gary Glitter et enfin, le principal, qui était tiré du générique de la série télévisée « *Doctor Who* ». Numéro 1.

[36]Certaines techniques de filtrages spécifiques (et notamment l'utilisation de filtres en peigne), permettent dans une certaine mesure de réaliser une forme de démixage. L'efficacité de celui-ci reste cependant soumise à la relative simplicité du contenu spectral de la réduction sonore à traiter (à savoir que les différents instruments doivent occuper des champs fréquentiels différents), excluant pratiquement toute forme de musique électroacoustique ou même de musique orchestrale (il reste en effet impossible à partir d'une réduction sonore de séparer spectralement différentes sources présentant des composantes fréquentielles identiques). Cf. [1] pour le projet *Semantic HIFI.*

[37]Ici dans le sens de l'utilisation de dispositifs de projection du son dans l'espace.

[38]Le fonctionnement des pistes de données se rapprochant alors d'un codage numérique des différents *gestes* de mixage et/ou de spatialisation appliqués aux pistes sonores de base. L'intérêt étant ici de pouvoir isoler tel ou tel élément au sein d'une écoute active et au final d'appliquer sa propre interprétation (de mixage et/ou de spatialisation) en *supplément* de l'interprétation initiale.

[39]On peut d'ailleurs ici imaginer que cette fonction soit assez facilement implémentable sur les outils de création sonore et de mise en forme temporelle par l'adjonction d'un *plug-in* de gestion du format et de création des pistes de données d'interprétation en sortie des

les diverses versions de l'œuvre présentes sur le support (une - ou, pourquoi pas, *plusieurs* - version(s) stéréophonique(s), une version 5.1, une version pour système de diffusion quadri- ou octophonique, ou encore binaurale pour la diffusion au casque, etc. assurant ainsi la compatibilité des différents systèmes de diffusion) mais aussi suivant la version dont l'auditeur pourra alors devenir l'interprète en fonction de ses propres choix esthétiques et selon la configuration de son matériel de diffusion personnel.

En effectuant des réductions intermédiaires, et en définissant par là même les caractères inaltérables de l'œuvre, le compositeur garderait bien entendu un contrôle sur le matériau[40] qu'il laisserait à disposition de l'auditeur pour de nouvelles interprétations. La plupart du temps réservé de nos jours aux seuls compositeurs et à leur entourage le plus proche, le support multipiste d'une oeuvre se devra cependant de fournir un matériau sonore le moins verrouillé possible[41] afin de permettre, dans le respect de l'identité de l'œuvre, des possibilités d'interprétations les plus riches. Telle est la voie pour le développement d'une véritable pratique d'interprétation du mixage et de la spatialisation, dont l'auditeur ne reste pour l'instant aujourd'hui que le témoin lorsqu'il est en mesure d'apprécier les différents mixages ou (re)masterisations réalisés à partir des bandes originelles de certaines œuvres musicales du répertoire populaire, ou lorsqu'il peut comparer différentes interprétations d'une même œuvre diffusée sur un orchestre de haut-parleurs.

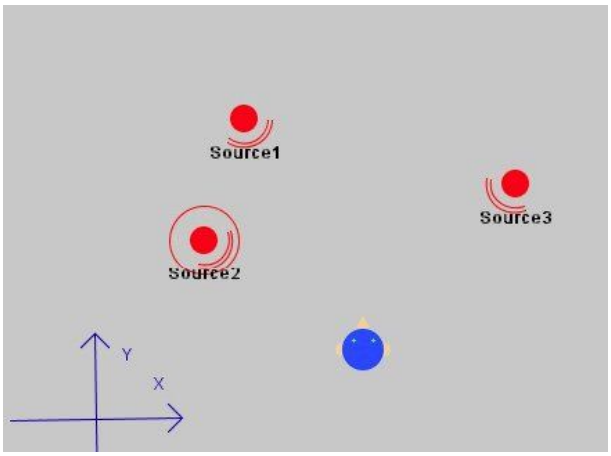


Figure 3. Exemple de *GUI* pour la spatialisation (source IRCAM - Projet *Semantic Hifi*)

L'interface de contrôle intégrée ou *agencée* au système de diffusion resterait encore à être définie précisément (*hardware*, sous la forme de potentiomètres linéaires équipant habituellement les tables de mixage, ou logicielle, sous la forme d'une *GUI - Graphical User Interface*). Elle se devra en tout état de cause de prendre, dans une convergence retrouvée des outils de la création et de la réception musicale, la même forme que celle des outils que le compositeur aura lui-même pu utiliser : telle est la condition de l'établissement de ce nouveau standard multipiste, des pratiques phonographiques dont il serait porteur, ainsi que de la véritable forme d'associativité qu'il apporterait au milieu de la phonofixation.

séquenceurs...

[40]Et ses secrets de fabrications...

[41]Ce qui est malheureusement encore trop souvent le cas de nos jours où les spatialisations ont pour matériau de base de simples réductions stéréophoniques, limitant les capacités d'actions de l'interprète spatialisateur (même si cette interprétation n'est pas à strictement parler une interprétation musicale, cf. sur ce point [8]).

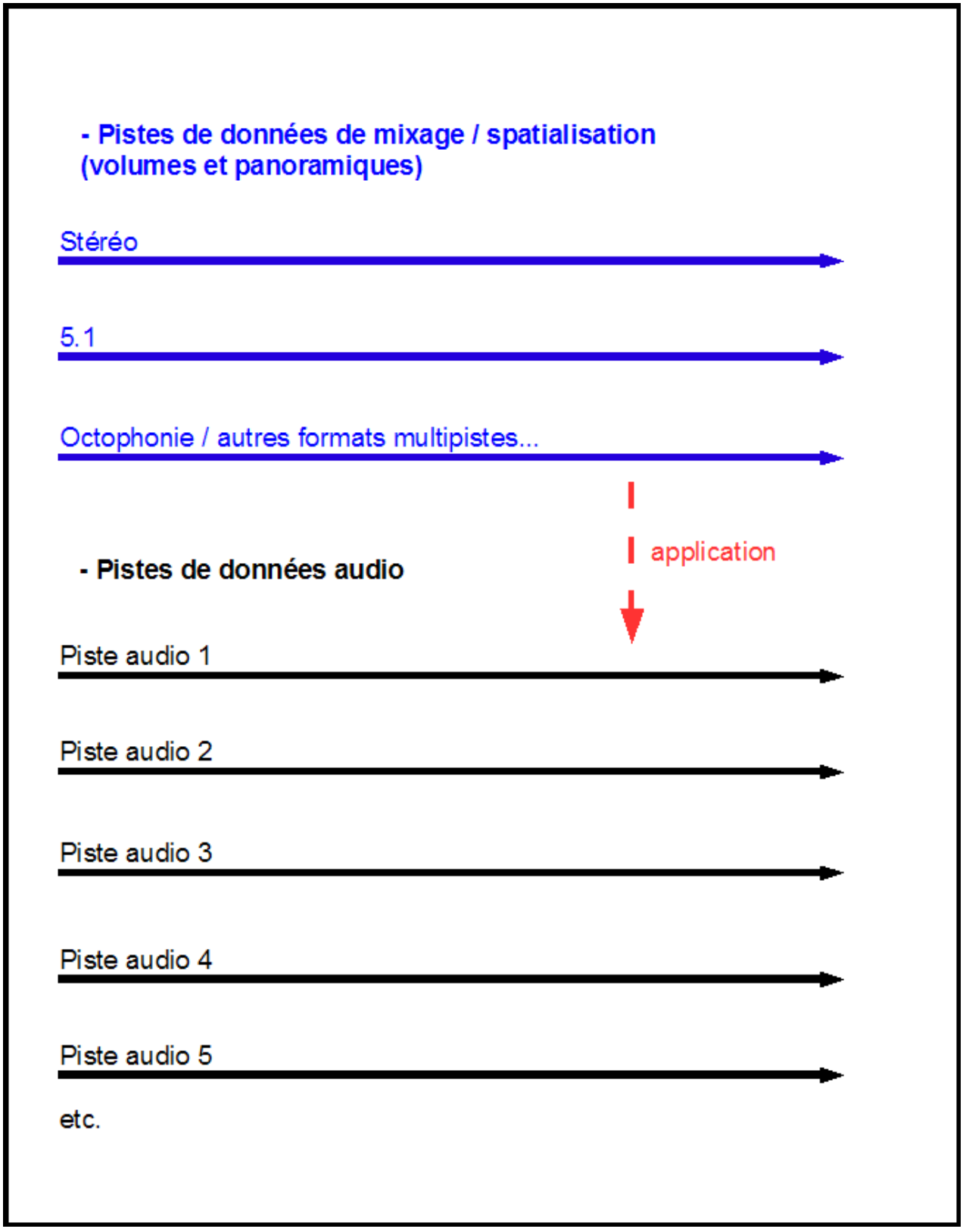


Figure 4. Format d'indexation verticale : schéma de principe

4. RÉFÉRENCES

[1] Ben-Shalom A. & Dubnov S. « Optimal Filtering of an Instrument Sound in a Mixed Using Harmonic Model and Score Alignment », Proceedings of International Computer Music Conference, Miami, 2004.

[2] Bricout R. *Les enjeux de la lutherie électronique : de l'influence des outils musicaux sur la création et la réception des musiques électroacoustiques*, Thèse inédite, Ecole Doctorale SHS, Université de Lille, 2009.

[3] Bricout R. « Les incarnations du *sampler* au XXe siècle : l'avènement du musicien-luthier », Publication des actes du colloque « Musique concrète, 60 ans plus tard – EMS08 », Electroacoustic Music Studies, Ina-GRM/MINT-OMF Paris Sorbonne, 2008, disponible en ligne *via* http://www.ems-network.org/ems08/papers/bricout.pdf

[4] Donin N. & Stiegler B. « Le tournant machinique de la sensibilité musicale », Révolutions industrielles de la musique, *Les Cahiers de médiologie*, n°18, Donin N. & Stiegler B. éds., Cahiers de médiologie/IRCAM, Fayard, Paris, 2004, p.6-17.

[5] Roussel R. *Locus Solus*, Flammarion, Paris, 2005.

[6] Stiegler, B. *De la misère symbolique, 2. La catastrophè du sensible*, Galilée, Paris, 2005.

[7] Tiffon V. « L'interprétation des enregistrements et l'enregistrement des interprétations : approche médiologique », *DEMéter*, Université de Lille, 2002, disponible en ligne *via* http://demeter.revue.univ-lille3.fr/interpretation/tiffon.pdf

[8] Tiffon V. Bricout R & Lavialle R. « Sortir de l'aporie du concert acousmatique par le jeu musical des arts de la sonofixation », *DEMéter,* Université de Lille, 2007, disponible en ligne *via* http://demeter.revue.univ-lille3.fr/manieres/edesac1.pdf

SYNTHESE SONORE, MODELES ET REPRESENTATIONS

Guillaume loizillon
Université Paris 8
Guillaume.loizillon@univ-paris8.fr

RÉSUMÉ

Les modes de synthèse par modèle physique sont maintenant bien établis dans les usages de l'informatique musicale. Depuis qu'ils sont disponibles comme sous ensemble de programme comme Max msp ou autres langages spécialisés, ils permettent de formaliser des accès et des interfaces visant à figurer des métaphores de cette *réduction* à la physique. L'usage du temps réel conduit à envisager des modalités de mise en œuvre spécifiques à travers des représentations et des figurations ayant pour modèle le dispositif même.

1. INTRODUCTION

L'usage de la synthèse par modèle physique fait appel à des paramètres singuliers qui souvent diffèrent de ceux des autres modes de synthèse. Plus qu'un simple changement de l'appareillage méthodologique c'est la conception même de l'acte musical qui s'engage dans cette utilisation. Les modèles de synthèse additifs soustractifs ou généralement tout ceux qui visent à la construction du signal ont en commun des éléments qui qualifient le son selon ses caractéristiques acoustiques, c'est à dire selon des effets perceptifs. Il est question ici de fréquences, d'intensités, de durées. Les paramètres peuvent définir des fréquences oscillations, des valeurs de filtre, des définitions d'enveloppe dynamique ou de hauteur, des tailles de grains ou encore des dosages de modulations. Les modèles physiques substituent à toutes ces données celles de dimensions physiques, matière, force ou énergie, c'est à dire celles de la causalité phénoménale des sons, qui affirme une dimension événementielle. Plus qu'une simple substitution de paramètres il s'agit de penser ces données comme de réelles opportunités expressives. Le terrain de l'image ou de la métaphore apparaît ici comme fécond, et nous développerons quelques propositions articulées sur ces modèles, sans négliger les possibilités de penser cela dans le cadre plus général de toutes les formes de synthèse sonore. Nous désirons pas installer de hiérarchie ni développer une taxinomie qualitative. La diversité des modalités d'accès à la synthèse des son est plutôt à considérer comme un ensemble des pertinences spécifiques en vue d'une création musicale ou sonore.

2. SUR LES MODELES DE LA SYNTHESE SONORE

En synthèse sonore, la fonction du modèle est double. Elle identifie tout d'abord les opérations d'ordre technologiques visant à la réalisation sonore. Comme dispositif théorique, le modèle s'articule dans des enchaînements de procédures que l'usage des langages informatiques rend apparent, bien plus que les instruments de musique électroniques du type synthétiseurs à clavier. Mais le modèle est également et peut-être avant tout un support de l'imaginaire dans un système où on été séparés tous les éléments qui conditionnent l'émergence du son musical : le principe acoustique, les modalités d'énergie mises en œuvre pour son excitation, les gestes instrumentaux qui les actualise. En balayant cet enchaînement de causalités, la synthèse peut consister en grande partie à reconstruire, même métaphoriquement, de tels mécanismes. Cela pose de manière concrète et opératoire la question des accès ou des interfaces quand on virtualise le processus dans le contexte de l'informatique. Ainsi, l'acte de composition débute dès le choix de mode de synthèse. Il s'actualise dans le vocabulaire même des procédures en tant que possibilité d'images fondatrices.

L'enchaînement de connexions oscillateur, filtre, générateur d'enveloppe est un schéma type qui, bien que ne relevant d'aucun principe mécanique acoustique, fonctionne de manière forte comme image d'une réalité sonore. Produit d'une historicité déjà robuste on en connait les différents avatars informatiques, directement construits en écho aux synthétiseurs instruments apparus dès les années 60, ces derniers figurant en quelque sorte le modèle instrumental original.

Les dispositifs de synthèse qui utilisent les modèles physiques, bien qu'en référence directe avec des objets du réel, sont paradoxalement en manque de représentations probantes et fondatrices dans le cadre de l'environnement informatique. Bien que de nombreuses aides graphiques existent, c'est d'autres types de représentations, même langagières, qui manquent à ces modèles.

Ainsi, l'utilisation musicale de la synthèse s'articule souvent plus en l'établissement de systèmes de représentations et d'intégration dans des contextes musicaux qu'à un seul usage tourné vers la recherche de sonorités originales.

2.1. Mises en œuvre au premier degré

Plusieurs procédures de modélisation physique sont disponibles au sein des différents langages informatiques musicaux. Ces modèles se caractérisent au premier abord par la singularité des paramètres mis en jeu en regard de ceux habituellement rencontrés dans les modes de synthèse de signal. Cette approche peu apparaître comme assez restrictive si l'on se contente simplement de penser la création sonore comme relevant seulement d'un réglage habile de ces paramètres. On retrouve ici les restrictions déjà formulées à l'encontre des modèles physique il y a quelques décennie par Jean-Claude Risset : « À première vue ils sont moins porteurs d'innovation que les modèles de signal, puisqu'ils restaurent dans l'ordinateur les limitations acoustiques ». [1]

Examinons deux archétypes de construction de dispositifs cartographiant quelques questions relevant de ce paradigme de la modélisation physique.

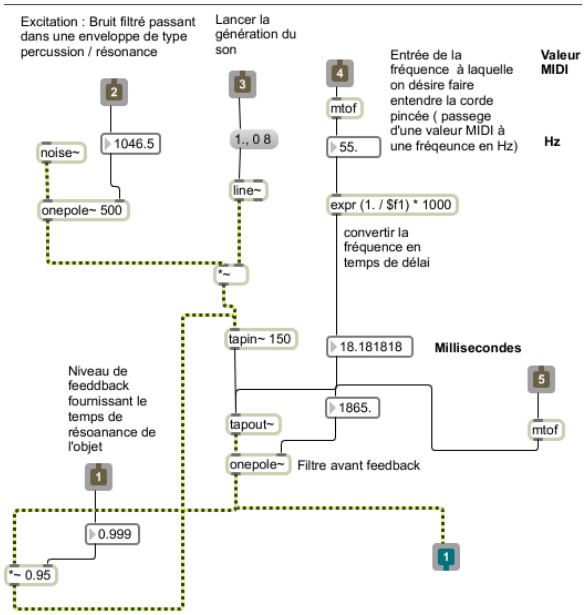


Figure 1. Configuration de base pour une mise en œuvre de l'algorithme de Karpus Strong

Ce schéma nous montre comment la formalisation d'un son de corde pincée peut passer par un processus de synthèse. Cet algorithme de modélisation physique, déjà ancien, nous semble emblématique. Il pourrait apparaître comme clos dans l'usage imitatif auquel il semble strictement destiné. Une exploration plus extensive de ses potentialités peut cependant encore donner des résultats tout à fait originaux et personnels.

On y remarque déjà une pensée sur le son scindée en deux entités distinctes : d'un côté celle du principe sonore et de l'autre celle des modes d'excitation. La génération du son est articulée autour de l'idée de fixer un temps de délai selon la fréquence que l'on désire obtenir[1]. On connecte à l'entrée de cette ligne retard un très bref stimulus sonore. En réinjectant dans le système même cet assemblage on obtient une mise en résonance à la hauteur précise que l'on a fixé. On en maitrise la l'extinction progressive en dosant ce pourcentage de feedback et en appliquant filtre passe bas juste avant cette réinjection.

Une rapide apparition disparition de bruit blanc (quelques millisecondes) figure un plectre, ou tout autre objet imaginaire, afin d'exciter le dispositif. Le filtre qu'on peut lui appliquer à ce son engendre un pincé plus ou moins dur et brillant. Passé la phase mimétique, tout un travail musical s'ouvre, en explorant de multiples modalités d'excitations : différentes enveloppes dynamiques, excitations par des sons acoustiques etc. Il est bien entendu intéressant de penser à une multiplication de lignes de retard, ouvrant par exemple à la possibilité de création d'espaces multi résonants

L'exemple suivant met en œuvre une modélisation d'objets physiques un peu plus avancée. Le patch écrit en en Max msp entre en relation avec un script extérieur par le biais de l'objet *modalys~* (développé à l'IRCAM). Ce script est généré par une programmation écrite au préalable en langage Lisp. Ce programme met en œuvre la bibliothèque et le moteur de synthèse modale, *Modalys*. En voici le listing complet qui s'achève par l'instruction de création du script utilisable depuis Max msp.[2]

```
(new)
;-------------------------------------------------------------
; Contrôleur permettant de faire varier la tension
; de la membrane depuis Max msp
;-------------------------------------------------------------
(defvar tension-ctrl)
(setq tension-ctrl
    (make-controller 'dynamic 1 -1 1500 "tension"))

;-------------------------------------------------------------
; Définition et paramétrage de la membrane
    circulaire
;-------------------------------------------------------------
(setq membrane (make-object 'circ-membrane
    (modes 90) (radius 0.7) (tension tension-ctrl)
    (freq-loss .07) (const-loss 0.5)))

;-------------------------------------------------------------
; Définition d'un point d'entrée sur la membrane du
    signal d'excitation depuis Max msp
;-------------------------------------------------------------
(defvar point-entree)
(setq point-entree (make-point-input 0 150))
```

[1] Formule de calcul du temps de délai : [(1/f)*1000] où f = la fréquence de vibration de la « corde » que l'on désire obtenir.

[2] Il est possible également d'utiliser un dispositif d'aide à la construction de script directement depuis Max msp, mais dans ce cas précis, nous sommes passé par la programmation en lisp et la conversion automatique en script.

```
(defvar entre-signal)
(setq entre-signal (make-controller 'signal 1 point-
      entree))
;-------------------------------------------------------------
; Connexion membrane signal
;-------------------------------------------------------------
(setq membrane-sig (make-access membrane (const
      .333 45) 'normal))
(make-connection 'force  membrane-sig entre-signal)
;-------------------------------------------------------------
; Point d'écoute sur la membrane
;-------------------------------------------------------------
(setq membrane-out (make-access membrane (const
      .21 90) 'normal))
(make-point-output membrane-out)
;-------------------------------------------------------------
; création du script pour exploitation depuis Max
      msp
;-------------------------------------------------------------
(save-script  (make-pathname-in-directory-of-source
      file "membrane.mlys"))
```

Le programme « fabrique » une membrane circulaire de 70 cm de rayon et de tension variable. Il définit différents dispositifs de contrôle permettant d'agir sur cet objet. L'excitation de la membrane se fait par application un signal en un point précis de sa surface. Le signal est interprété comme une force exprimée en Newton, variant à la fréquence d'échantillonnage.

Un développement du modèle consisterait à diversifier les modes d'excitations : frotter, frapper, souffler etc. L'exécution de ce programme ne produit pas de résultat sonore (d'autres instruction seraient nécessaires) mais génère dans sa dernière ligne le script qui sera directement adressable depuis l'objet *modalys~* dans un patch Max msp. Comme on le voit, on peut se représenter cette première étape comme la figuration d'un dispositif instrumental destiné à être mis-en œuvre. Ici réside l'originalité, sans doute radicale, de ce mode de synthèse.

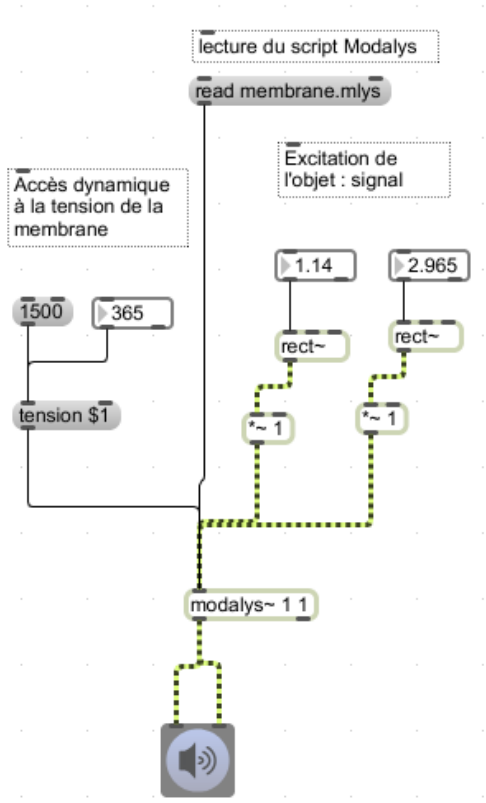


Figure 2. Patch Max msp permettant l'exploration en temps réel du programme Modalys.

Le patch reproduit ici, présente un aménagement minimal qui permet de faire fonctionner le script issu de Modalys. Au delà de la mise en œuvre pratique, s'ouvre alors un territoire complexe qui vise à développer un pensée musicale autour d'un tel dispositif et de ses extensions. S'il est à l'évidence possible d'oublier le modèle dans son aspect mathématique et la virtualité qu'il met en acte, il semble en revanche que la métaphore et l'image ne peuvent complètement s'absenter d'une pensée musicale qui se construit à partir d'un tel système. Le vocabulaire même y conduit : il s'agit non pas de construire un phénomène sonore depuis ses composantes fondamentales mais d'imaginer des modes d'excitations d'un objet : de figurer des jeux d'énergie, de penser le son comme geste instrumental, à considérer l'instrument comme *réseau de contraintes* comme l'affirme Mauro Lanza [2]. Tout comme le paradigme de la synthèse granulaire dépasse le seul énoncé de son schéma initial, la concaténation de molécules sonores, le modèle physique (et sans doute tout mode de synthèse) conduit à des formalisation spécifiques et donc à des stratégies spécifiques de composition. Un seul usage *timbral* risquerait de rendre anecdotique ces catégories sonores s'il n'était prolongé par des considérations sur l'implication de la typologie du mode synthèse sur l'acte même de composition.

2.2. De l'énergie comme modèle esthétique

Ainsi, la question des modèles physiques, pour peu que l'on désire en penser un usage original, s'engage du coté de la représentation métaphorique du geste et de l'énergie. Le paradigme des musiques électroacoustiques s'est souvent trouvé dépassé par d'autres questions que celles liées à la recherche de sons nouveaux. Si composer des timbres reste un horizon, l'usage montre que ce seul objectif ne peut circonscrire la totalité de la pratique. La mise à plat de la chaine causale de la production du son musical que ces musiques impliquent, conduit à une prise en considération d'autres valeurs que celles de la création de timbres.

La figuration des mouvements d'énergie dans une perspective esthétique où viennent s'agréger des formulations et des images d'une certaine recomposition utopique nous apparaît comme un horizon fécond pour l'usage des modes de synthèse, particulièrement ceux qui implique une modélisation physique. Comment instrumentaliser alors ces données de la synthèse ?

En exploitant les modes d'excitation et des configurations d'objets de plus en plus complexes, on obtient un ensemble sonore que l'on peut qualifier d'événements, dans le sens ou le processus génératif introduit par l'utilisation d'une figuration de l'énergie devient une part active de l'écoute. Il est également possible de rendre réelle l'idée d'une recomposition utopique d'instruments ou d'objets : des dimensions improbables, des modes d'excitations impraticables, des effets de matières qui tout en maintenant un lien de plausibilité avec le réel en font entendre des configurations chimériques.

C'est pour cela que le lien avec la réalité physique qui peut sembler au départ une limitation en regard des modes de synthèse de signal devient dans ce cas une richesse dans la mesure ou il institue dans l'imaginaire auditif un statut à la causalité des sons, avec ce paradoxe intéressant que cette causalité devient à son tour une donnée paramétrable, susceptible d'être formalisée par une écriture.

La formalisation ne peut s'organiser en dehors des prédicats du modèle. C'est à dire : c'est en regard de la représentation des éléments que met en œuvre le mode de synthèse qu'il s'agit d'imaginer des accès, des interfaces et des images pour lier le tout. La structuration même de l'écoute, quand bien même celle-ci ne passerait que par l'acousmatique est engagée dans tout dispositif de création sonore. L'imaginaire qui s'y déploie nécessairement demeure un fondement et un axe de composition. Ainsi dans sa part procédurale le modèle physique restitue à la nature sonore sa dimension causale et événementielle même s'il est possible de la considérer dans sa phase de calcul comme une forme d'interface particulière pour une synthèse additive.(Situation surtout vraie pour la synthèse modale)

Pourtant, pour peu que l'on adopte une attention plus phénoménologique, la synthèse modale ouvre des perspectives vers des événements sonores singuliers. « L'effet de réel » prend ici une dimension toute particulière et fondatrice ; l'obtention d'un rendu hyperréaliste, si l'on admet ce terme en analogie avec ce mouvement pictural des années 70 ou les représentations glacées et distanciées procurait au spectateur un sentiment d'étrangeté dû au mode de rendu du réel. La synthèse modale institue dans la perception l'idée de matière, d'énergie et d'interactions entre des objets animés par la force. Dans ce cadre il devient tout aussi important de concentrer l'opération musicale non seulement sur un travail strictement de timbre mais également sur des propositions sonores mettant en musique ces jeux de forces.

3. POUR UNE ESTHETIQUE DE LA SYNTHESE

Le son issu de la synthèse sonore maintient une relation avec le réel en dépit d'une apparente autonomie de structure. Cette étrangeté radicale en constitue la force, celle qui conduit à la spécification d'une invention. On sait bien que la synthèse n'a pas de véritable vocation mimétique. L'imitation de timbre acoustique ne peut se définir comme son objectif, bien que cette perspective ne soit pas sans intérêt dans l'optique d'une compréhension du phénomène du son musical. La mobilité, la fluidité, la plasticité sont les qualités reconnues de la synthèse sonore. Pourtant, au-delà de ces données c'est la figuration d'une dynamique, d'un ancrage événementiel qui apparaît comme une voie pour la composition.

Cette construction d'effets de réels se rencontre dans les musiques populaires qui font usage intensif de l'électronique. La nature machinique y devient graphique et productrice de fictions et d'images. Des musiques « planantes » des années 70 à la mosaïque du mouvement techno ou electro actuels on ne peut considérer leur rendu sonore comme indépendant de l'imagerie qui se construit en parallèle des dispositifs de fabrication ; ordinateurs, synthétiseurs, boites à rythmes, mais aussi la masse imposantes des « Sound systems », l'imaginaire organique qui se dégage des câblages et des connections même quand celles-ci demeurent virtuelles.

Un mode de synthèse, par delà ses performances propres à offrir une palette variée de sonorités, fonctionne également comme un stimulant de l'imaginaire, comme une base scénaristique. Le mode de synthèse est en ce sens une figuration de différentes réalités autant qu'une ensemble de procédures.

3.1. Chemins de l'inouï

Ce qui n'a jamais été ouï : tel pouvait sembler être une partie du programme des initiateurs des musiques concrètes ou électroniques de la fin des années 40. Pourtant, une telle volonté semble avoir dû,

restreindre ses ambitions, ou du moins les recentrer vers une relation plus dialectique en considération des différentes dimensions de l'écoute. Il s'avère indispensable de lier l'attachement au son comme effet sensoriel à celui de sa valeur de support d'information et de sens. Ainsi, le timbre comme valeur isolable, indépendante de toute relation causale, même la plus indirecte, apparaît comme difficilement atteignable, sans doute à mettre en relation avec l'impossible pensée d'un être fantastique qui ne soit pas composé avec des éléments d'êtres réels.

Entendre apparaît de ce fait comme un acte indéfectiblement lié à l'élucidation et à la reconnaissance. L'étrangeté radicale, en matière sonore, apparaît comme une aporie tant une relation événementielle est liée au phénomène sonore, y compris dans la musique qui se propose de faire jouer le son et l'écoute pour eux-mêmes, dans une valorisation de la sensorialité sonore .

Ainsi, la musique de la seconde moitié du XX^{ème} siècle a hautement développé des techniques et des esthétiques qui se réclament d'une création fondée une certaine figuration de l'inouï. Elle agit sur le plan pratique soit par transformation de sons existants, soit en court-circuitant tout passage par l'acoustique mécanique avec les différents modèles de synthèse. Tout cet ensemble a paru annoncer l'avènement d'une catégorie de sons sans précédents ni référents et posant à la perception auditive les défis les plus fondamentaux. Il a pu apparaître que la voie était ouverte à un monde sonore qu'aucune contrepartie issue du *monde réel* ne déterminerait. La situation acousmatique qui instaure une écoute sans relation à la cause n'a cependant pas pleinement institué ces mondes sonores, littéralement dégagés de toute nécessité d'identification. Même les *images de sons* tels que François Bayle les formalise ne semblent pas échapper à cette relation dans la mesure où précisément elles doivent faire image.[3] Cette sujétion au réel a au contraire permis de mettre l'accent sur la multitude des relations au monde présentes dans chaque événement sonore, même le plus engagé dans une pensée musicale immersive ou sensorielle.

Quelques décennies de pratiques nous montrent que le son envisagé dans un contexte musical apparaît comme difficilement affranchi de toute relation causale. Le dispositif instrumental assure par la présence de l'objet et des gestes qu'il implique le lien de causalité en adéquation avec ces gestes et un certain caractère qui demeure figuratif. Les procédures de l'électroacoustique interrogent l'ensemble de ces déterminations. Ainsi, il s'agit de discuter le fait qu'une rupture radicale avec toute évocation causale ne peut intégralement s'envisager, et que les mécanismes de l'écoute agissent toujours dans le sens de l'institution d'une figuration. Un objectif artistique à suivre est d'explorer cette proposition avec les différentes modalités de création sonore que développent les techniques d'analyse de synthèse et de transformation des sons. Il devient alors possible de poser l'hypothèse que la métamorphose de l'écoute se fonde sur la territorialisation du sonore dans le champ du réel.

4. CONCLUSION ET PERSPECTIVES

La pensée pionnière des musiques électroacoustiques s'est affirmée comme celle qui allait ouvrir la musique à un océan de sons, considéré comme un milieu naturel qu'il est possible de modeler Elle instrumentalise une pensée sonore qui affirme la capacité d'abstraire l'écoute de toute évocation d'un principe causal, de tout système langagier. Pourtant, la musique qui fait appel à la synthèse sonore, comme celle des sons transformés, ne peut totalement se dégager de cette idée que c'est le dispositif et la construction du sonore qu'il engage qui constituent et déterminent l'acte musical, que celui-ci passe par le jeu ou par la composition. Le son sans cause, celui qui parle directement à la pensée reste de domaine de l'improbable. On en soupçonne même ses débords inquiétants. Tout son est, semble t-il, indéfectiblement attaché à ce fait d'être le son de quelque chose, tout en affirmant un statut sensoriel propre et autonome. C'est à la création musicale qu'il revient de jouer sur ces frontières.

5. REFERENCES

[1] Risset Jean-Claude, *Synthèse et Matériau Musical*, Les cahiers de l'IRCAM, la synthèse sonore, 1993

[2] Lanza Mauro, *Retour du refoulé. La contrainte instrumentale en milieu informatique*, Révolutions industrielles de la musique, Cahiers de médiologie 18, Fayard, 2004

[3] Bayle François *Musique acousmatique : propositions... positions*, Buchet/Chastel, Paris, 1993

SUM: DE LA SONIFICATION D'IMAGE À LA COMPOSITION GRAPHIQUE ASSISTÉE PAR ORDINATEUR

Sara Adhitya
IUAV, EHESS, STMS Lab
(IRCAM/CNRS/UPMS)
sara.adhitya@ehess.com

Mika Kuuskankare
Sibelius Academy,
CCRMA, Stanford University
mkuuskan@siba.fi

RÉSUMÉ

Cet article porte sur le développement de l'outil SUM, une bibliothèque utilisateur avec une interface graphique au sein de l'environnement de composition assistée par ordinateur PWGL, visant à l'intégration de l'image et du son. Nous allons détailler sa structure interne, constituée de couches d'images, de convertisseurs et de chemins. Nous allons présenter le processus de sonification, l'extraction des données graphiques et leur traduction dans les paramètres audio. Enfin, nous discuterons des applications possibles de SUM, de la sonification d'image à la composition assistée par ordinateur, résultant de cette structure.

1. INTRODUCTION DE L'OUTIL SUM

L'outil SUM permet l'intégration de l'image et du son a travers une interface graphique. Créé à l'origine pour la sonification de cartes utilisées dans la planification urbaine [1], il prend en charge l'importation et la création de couches d'images multiples (pixels et vecteurs) en tant que données d'entrée. Les données sont ensuite extraites par le dessin d'un ou plusieurs tracés vectoriels sur les zones d'intérêt, puis l'association des attributs graphiques et des attributs sonores produit une partie audio. Ainsi SUM adopte une approche spatio-temporelle et multidimensionnelle de la sonification d'image, ce qui le distingue des autres outils de sonification d'image tels que SonART [2].

Comme une librairie dans PWGL [3], environnement visuel de composition assistée par ordinateur basé sur LISP, SUM peut aussi être utilisé comme un outil de composition graphique. ENP [4], l'éditeur de notation musicale interne de PWGL, permet strictement la description de partitions graphiques à base d'objet, plutôt que l'exploration pixel-par-pixel d'une partition comme une image. D'autres environnements graphiques de composition assistée par ordinateur, tels que HighC [5] et Iannix [6], inspiré par le système UPIC de Xenakis, permettent le dessin d'objets graphiques, mais sont limités à une ligne temporelle unique et horizontale. Cependant SUM, avec sa capacité de créer et de lire des objets le long de plusieurs chemins spatio-temporels, permet à l'image d'être composée et jouée comme une partition graphique et ouverte, à partir de multiples perspectives.

Cet article va discuter de la structure de SUM, qui propose une approche multidimensionnelle de la sonification d'image et de la composition assistée par ordinateur.

2. LA STRUCTURE INTERNE DE SUM

L'outil SUM se compose de trois éléments principaux: image, chemin, et convertisseur. La section suivante explique chacun de ces éléments et leurs interrelations.

2.1. Images

SUM utilise des images comme sources de données. Chaque image est décrite par une table de couleurs, dans laquelle chaque couleur d'intérêt est associée à une valeur numérique arbitraire, pour être référencée dans le processus de sonification. SUM permet la superposition de plusieurs images, ce qui permet la synthèse de l'information graphique, visualisable comme une matrice 3D de données comme le montre la figure 1. Un groupe de sources de données est appelé un *dataset*, à partir duquel un certain nombre de couches d'image peuvent être sollicitées en tant que sources de données dans le processus de conversion.

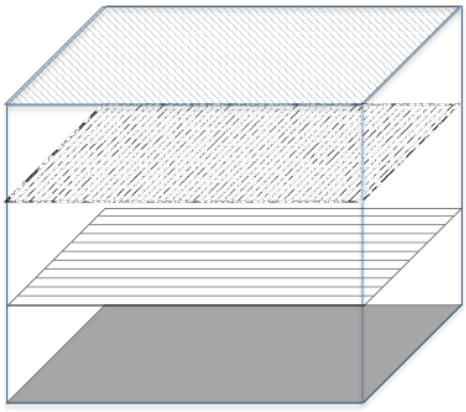


Figure 1. La visualisation d'un '*dataset*' comme une matrice 3D, composée des couches d'images 2D

SUM permet la coexistence d'images pixellisées et d'images vectorielles. La flexibilité de l'importation en pixels permet à toute visualisation, y compris celles produites par d'autres logiciels, d'être sonifiées. La capacité de l'outil de dessin vectoriel permet d'être utilisé comme un outil de design assisté par ordinateur, tels que Adobe Illustrator ou AutoCAD, et donc de faire des dessins et des changements graphiques directement dans le logiciel.

2.2. Convertisseurs

Un convertisseur définit les caractéristiques du résultat sonore du processus de conversion. Il traduit les attributs graphiques extraits de l'image en événements audio discrets, en définissant les attributs sonores de hauteur, volume, articulation et timbre. La définition de chaque attribut sonore est indépendante de l'autre. Ainsi, un convertisseur peut se référer à de multiples sources de données. (figure 2) Un groupe de convertisseurs s'appelle un *mapper-group*.

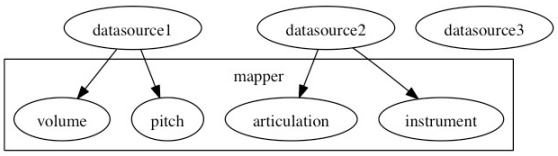


Figure 2. Le convertisseur de SUM : une définition possible des attributs sonores à partir de source de données

2.3 Chemins

Un chemin définit la connexion entre l'espace graphique et le temps musical. Il est un objet spatio-temporel constitué des qualités suivantes: la position et direction, le délai, la durée, et la vitesse. Le chemin est une ligne dessinée par l'utilisateur sur la zone d'intérêt, à laquelle sont attribués une vitesse et un délai. SUM permet la coexistence de plusieurs chemins de vitesses et délais différents.

3. LE PROCESSUS DE CONVERSION EN SUM

Le processus de conversion en SUM, de l'image au son, est un processus en deux étapes: les données graphiques sont récupérées à partir d'une source de données, selon un chemin ; il est ensuite appliquée à un convertisseur pour sa transformation en attributs sonores.

3.1. Extraction de données

Le processus de conversion SUM est basé sur le chemin. Les données sont récupérées à travers le dessin d'un tracé vectoriel sur une image, et l'échantillonnage de l'image dans cette voie. Le trajet vectoriel est tramé conformément à l'algorithme de ligne de Bresenham, afin de le décomposer en points d'échantillonnage discrets, tout en conservant l'ordre des points pour déterminer la direction du trajet le long duquel le temps passe. Ainsi, pour une ligne s'étendant vers le haut et la gauche, les pixels seraient échantillonnés dans l'ordre indiqué dans la figure 3.

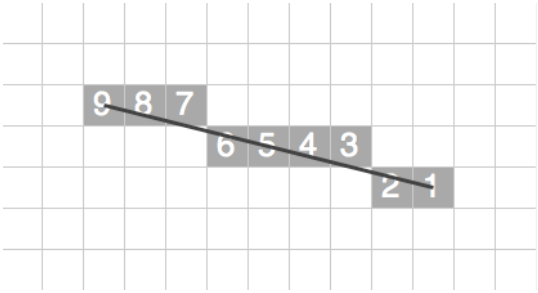


Figure 3. Schéma de l'algorithme de la ligne de Bresenham, montrant l'ordre d'échantillonnage

Chaque image est ensuite échantillonnée pixel par pixel afin de récupérer les données d'intérêt pour chaque point d'échantillon le long du chemin. L'utilisateur définit le début et la vitesse de la lecture pour déterminer la structure temporelle du processus de conversion.

3.2. Conversion des paramètres

Après l'extraction de l'information graphique le long d'un chemin, ces valeurs peuvent être appliquées à un convertisseur afin de générer les attributs sonores désirés d'un signal acoustique (la hauteur, le volume, l'articulation et le timbre). Le processus de conversion de paramètres est défini par une légende, à partir d'une source de données, avec une valeur sonore. Ceci peut être mis en œuvre, soit directement via l'interface graphique, ou avec une description en Lisp pour les conversions plus complexes.

L'application d'un chemin à un convertisseur produit un ensemble de paramètres sonores, qui peuvent ensuite être utilisés pour piloter une grande variété d'instruments, internes ou externes. PWGL a son propre synthétiseur interne ainsi que MIDI et OSC. Cela permet la connexion à des logiciels externes de synthèse sonore, tels que Max/MSP, et des possibilités flexibles pour la sortie audio.

Il convient de noter que le chemin et le convertisseur sont indépendants les uns des autres en termes de source de données. Ainsi différents convertisseurs peuvent être générés à partir du même ensemble de sources de données.

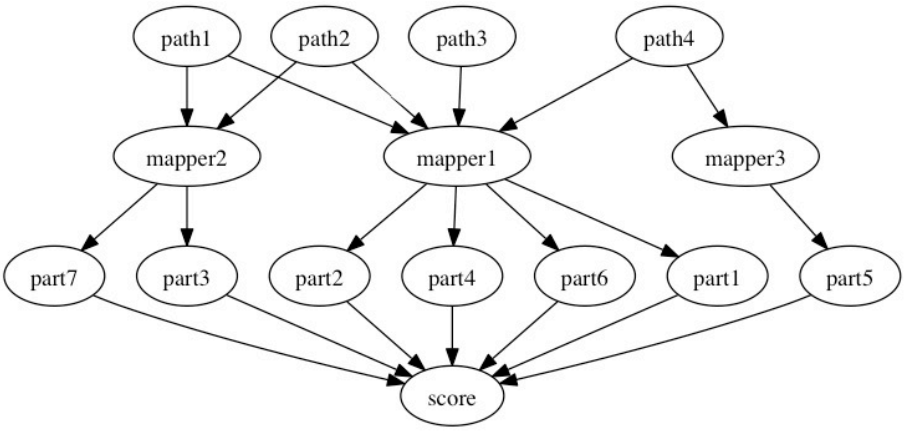


Figure 4. Un exemple d'une partition SUM- un réseau possible de chemins et de convertisseurs

4. PROCESSUS DE COMPOSITION EN SUM

Cette section portera sur le processus de composition en SUM. Ici, nous introduisons le concept de la partition SUM, composée de plusieurs parties.

Une partie SUM est une séquence d'événements audio, les qualités de ce qui est défini par l'extraction de données à partir d'un chemin et d'une image, et l'application de ce chemin à un convertisseur. Ainsi la génération d'une partie SUM est un processus basé sur le chemin. L'application de plusieurs chemins à convertisseur va produire des parties SUM multiples de la même qualité de timbre, mais de structure temporelle variable. L'application du même chemin à plusieurs convertisseurs va produire des parties SUM multiples de la même qualité spatio-temporelle, mais de qualités de timbre variables. Différentes combinaisons de chemins et de convertisseurs permettent la génération de parties SUM nombreuses à partir du même ensemble de données. La figure 4 montre un réseau possible de chemins et de convertisseurs, produisant une partition SUM.

5. APPLICATION : SONIFICATION D'IMAGE

La flexibilité du processus de conversion établi entre l'image et le son, a le potentiel pour une application à la sonification d'image – soit scientifique, soit artistique.

5.1. Sonification des données graphiques

L'outil SUM, avec son entrée à base d'images, et son processus de conversion défini par l'utilisateur, permet la sonification d'une image codifiée par la couleur. Cela en fait un outil flexible pour l'analyse des données basée sur l'image, car les résultats graphiques d'une analyse effectuée précédemment peuvent être convertis en fonction de son code couleur, sans problèmes de format de données ou de compatibilité de logiciels. L'extension de l'outil à des couches multiples d'images signifie que les données spatiales peuvent être synthétisées par sonification, même lorsqu'elles sont limitées par la visualisation. Basé sur le chemin, SUM est le plus utile outil d'analyse de séquences linéaires d'événements spatio-temporels. Un domaine d'intérêt est la planification urbaine (figure 5), dans lequel les cartes peuvent être sonifiées le long des chemins de parcours, afin de comprendre leur organisation spatio-temporelle à une certaine vitesse.

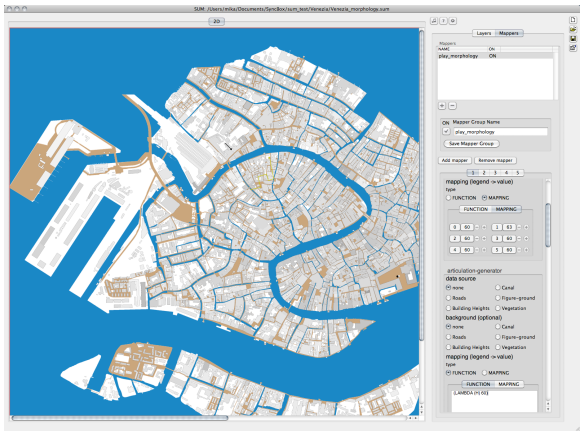


Figure 5. Sonification du plan urbain de la ville de Venise

5.2 Sonification artistique de 'musique visuelle'

Une autre application de la sonification d'image est dans le domaine des arts visuels, et en particulier la 'musique visuelle' des artistes comme Wassily Kandinsky [7] et Piet Mondrian. Avec SUM, on peut explorer cette notion de composition visuelle à travers leur sonification.

5.2.1 Sonification des chemins de mouvement

Kandinsky a exploré la ligne comme l'élément principal du mouvement en plusieurs études [8]. En SUM, on peut explorer le potentiel de ces 'lignes musicales' pour la structuration du temps dans une

partition graphique. Sur la figure 6, on a utilisé une étude de ligne de Kandinsky en tant que base, pour produire une composition sonore de différents mouvements spatio-temporels.

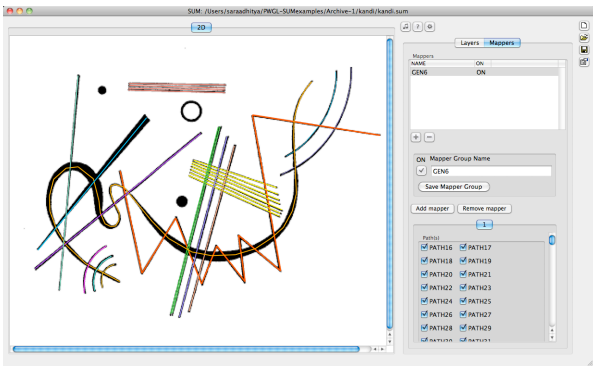


Figure 6. Une composition sonore basée sur les 'lignes musicales' de Kandinsky [9]

5.2.1 Sonification de 'compositions de couleurs'

La composition de couleur dans l'espace est une autre technique artistique de génération de rythmes visuels. Piet Mondrian a produit une série de tableaux intitulée «Composition», dans laquelle il explore la composition spatiale des couleurs primaires rouge, bleu et jaune. Dans son oeuvre *Woogie Broadway Boogie* (1942-43), le rythme du «boogie woogie» a été exprimé à travers ces couleurs le long d'une structure carrée ressemblant à des rues de New York [10]. Par le « mappage » de chaque couleur à un son différent, avec SUM on peut jouer cette peinture comme une partition graphique le long de ces chemins, et entendre le résultat rythmique (figure 7).

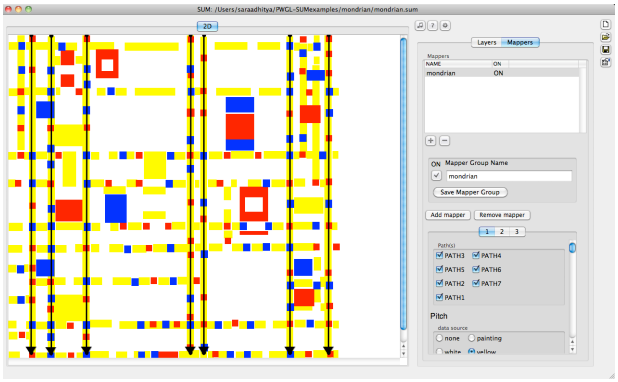


Figure 7. La sonification de *Broadway Boogie Woogie* de Mondrian [10]

5.3 Potentiel de sonification pour la composition musicale

À travers la sonification des oeuvres visuelles dans SUM, on peut explorer l'application de techniques de composition visuelle à la composition musicale. Dans la prochaine section, on va montrer l'utilisation de SUM comme outil de la composition assistée par ordinateur, vers la création d'une partition graphique.

6 APPLICATION : COMPOSITION ASSISTÉE PAR ORDINATEUR

L'outil SUM, avec sa capacité de dessin vectoriel, prend également en charge la création de partitions graphiques. Le processus de conversion défini par l'utilisateur, signifie qu'un compositeur est libre de créer son propre vocabulaire « graphique-son ». Il permet la création d'une partition de plusieurs couches graphiques (i.e. multiples dimensions spatiales), et sa lecture à partir de n'importe quelle direction, temps et vitesse (i.e. multiples dimensions temporelles). Cela donne la possibilité de génération d'une partition graphique, automatisée, ouverte et en 3D.

6.1 Création d'une partition graphique en SUM

Dans cette section on explique le processus de création d'une partition graphique dans SUM. Comme exemple on va recomposer la partition graphique de Rainer Wehinger (figure 8), de Artikulation (Gyorgy Ligeti).

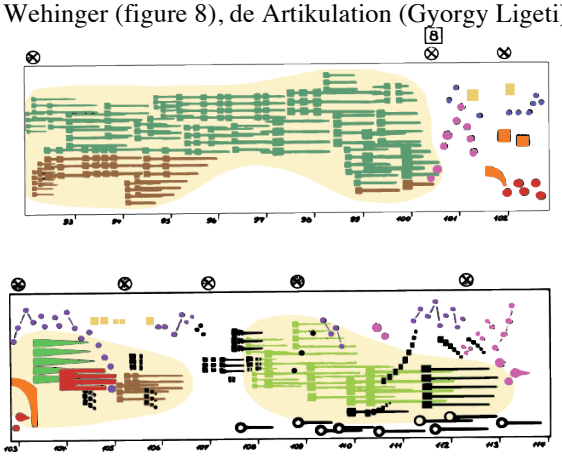


Figure 8. Une partie de la partition graphique de Artikulation (Ligeti) de Rainer Wehinger [11]

6.1.1 La table des couleurs des objets sonores

Wehinger a représenté graphiquement les différents objets sonores de Ligeti, en utilisant différentes formes et couleurs selon la légende de figure 9.

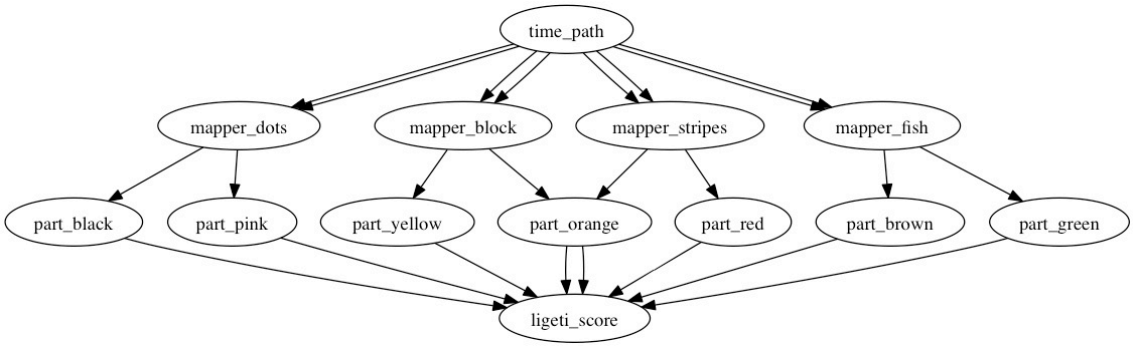


Figure 10. La structure d'une partition possible de Artikulation en SUM

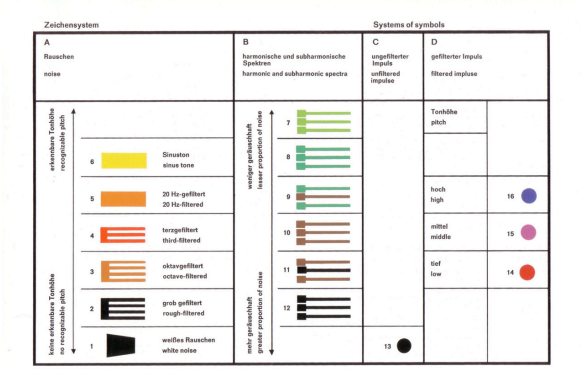


Figure 9. La légende de Wehinger de Artikulation [11]

A partir de cette catégorisation des objets sonores, on peut structurer une partition SUM comme représenté en figure 10.

6.1.2 La génération des couches d'images

Du fait que des objets sonores différents soient lus suivant les couleurs dans SUM, on peut construire les différents couches graphiques (figure 11) par rapport à leurs différentes couleurs.

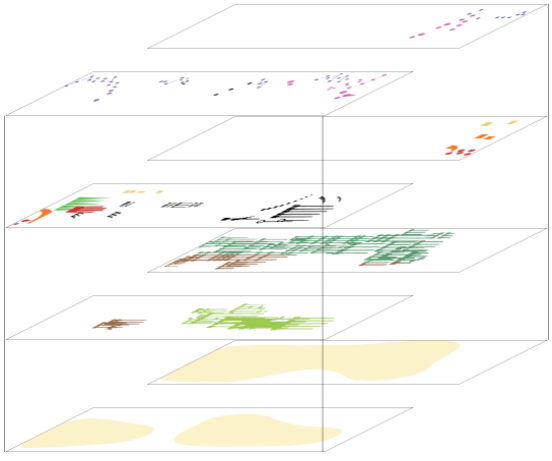


Figure 5. Les différents objets sonores d'Artikulation sont divisés en couches

6.4 La lecture de la partition graphique en SUM

SUM permet la coexistence de plusieurs chemins d'accès à des vitesses différentes, et donc il soutient la lecture d'une image comme une partition ouverte graphique. Dans la figure 12 nous donnons des exemples des différentes manières de jouer notre partition graphique «recomposée».

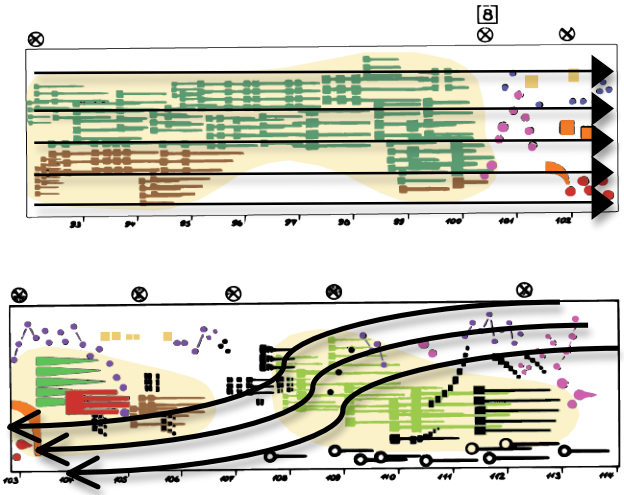


Figure 7. Des lectures alternatives de Artikulation

Au lieu de sa lecture 'conventionnelle', linéairement et de gauche à droite (dessus), on peut maintenant la jouer avec des chemins à partir de plusieurs directions (en bas), soit synchronisées, soit à différentes vitesses ou avec des différents temps de commencement.

6. CONCLUSIONS

Comme on le voit ci-dessus, la structure de l'outil SUM permet l'intégration de l'image et du son en temps, dans de multiples dimensions spatiales et temporelles. À partir de l'objectif de sonifier des plans urbains, pour l'analyse et la conception urbaines, il peut également être utilisé dans la composition d'une partition musicale graphique et multidimensionnelle. La structure flexible de SUM s'applique à la représentation audio-visuelle de systèmes complexes en général, ce qui nous permet de représenter plus que la simple somme de leurs parties, mais les relations

spatio-temporelles entre eux.

Les améliorations futures incluent l'automatisation de l'extraction de la palette couleur de l'image, et donc la génération de la table de couleurs. Nous visons également à améliorer notre approche à l'échantillonnage de chemin, afin de déterminer plus précisément la durée d'un chemin.

7. REFERENCES

[1] Adhitya S., Kuuskankare M., "The Sonified Urban Masterplan (Sum) Tool : Sonification For Urban Planning And Design", In: *17th ICAD International Conference on Auditory Display*, (2011).

[2] Woon Seung Yeo, Jonathan Berger, Zune Lee, "SonART: A framework for data sonification, visualization and networked multimedia applications", In: *30th ICMC International Computer Music Conference*, (2004)

[3] Laurson M., Kuuskankare M., Norilo V.: "An Overview of PWGL, a Visual Programming Environment for Music", In: *Computer Music Journal*, vol. 33, no.1, pp.19–31 (2009)

[4] Kuuskankare M., Laurson M., "Expressive Notation Package", In: *Computer Music Journal*, 30(4), pp. 67–79 (2006)

[5] Baudel, T., "High C draw your music", http://highc.org/history.html

[6] IanniX, a graphical open-source sequencer, based on Iannis Xenakis works, for digital art http://www.iannix.org/en/index.php

[7] "Visual music" – term first used by art critic Richard Fry in 1912 to describe Kandinsky's artwork

[8] Kandinsky, W. *Point to Line and Plane*, Dover Publications, Inc., New York (1979)

[9] Based on a line composition by Kandinsky (1979:165)

[10] Based on the painting by Piet Mondrian, Broadway Boogie Woogie, MOMA, NY (1942-43)

[11] Rainer Wehinger's graphic score (1969) for Gyorgy Ligeti's Artikulation (1958) http://www.tumblr.com/tagged/artikulation

DE L'INSTRUMENT ACOUSTIQUE A L'INTERFACE GESTUELLE, PARCOURS DE L'INTERPRETE CREATEUR

Barah Héon-Morissette
IACT - Institut Arts Cultures et Technologies, Université de Montréal
barah.heon-morissette@umontreal.ca

RÉSUMÉ

L'interprète est un spécialiste du geste instrumental et de la scène. Étant percussionniste et compositrice, c'est dans cette optique que j'ai entrepris le développement d'une interface gestuelle correspondant à ma corporéité.

Cette interface nommée, SIC - Scène Interactive pour la Créativité - est un système ergonomique permettant aux créateurs d'explorer et de développer un geste expert dans une dimension encore peu exploitée dans le domaine musical : le geste-son dans l'espace. Elle exploite des technologies de captation du mouvement et est constituée de deux composantes : kinKI utilisant le périphérique Kinect et un sol sensible captant la pression exercée par l'interprète en différents points de la surface.

Le développement d'une interface gestuelle appelle un questionnement sur le geste instrumental, la valorisation de la pratique, les différents aspects reliés à l'ergonomie ainsi que le rôle de cette dernière et de la transmission sur le plan de la pérennité des œuvres.

1. INTRODUCTION

La vision créatrice est le moteur de toutes les démarches entreprises et présentées dans cet article. L'interaction entre ma vision d'interprète et de compositrice a influencé la recherche et le développement de l'interface gestuelle SIC.

Le corps étant au centre de mes préoccupations d'interprète, je définis les gestes associés à ma corporéité ainsi que les paramètres essentiels à la valorisation de la pratique qui seront garants de la pérennité des œuvres utilisant le geste-son dans l'espace.

2. CHOISIR UN INSTRUMENT

2.1. Instrument acoustique

Dans notre société occidentale, choisir un instrument de musique est une décision personnelle, parfois issue d'un coup de cœur, d'une expérience sonore musicale intense et marquante [1]. Dans le cas de l'instrument acoustique, il est majoritairement un choix datant de l'enfance ou encore de l'adolescence. Le changement d'instrument dans un parcours musical peut survenir et être dû au développement physique de l'enfant par exemple le passage de la flûte à bec vers la flûte traversière.

La physiologie de l'instrumentiste est un facteur déterminant quant à ce choix. De plus, le sentiment d'appartenance pour son instrument est, pour l'interprète, un signifiant touchant à la longévité de sa pratique instrumentale professionnelle ou amateur. En effet, la pratique quotidienne nécessaire à la réussite est une activité qui se répétera tout au long de sa vie ; il est donc essentiel que l'instrumentiste se projette dans l'avenir avec son choix instrumental.

2.2. Interface gestuelle

Pour la génération de musiciens actuels[1], choisir une interface gestuelle pour une pratique instrumentale est un acte résultant d'une réflexion, d'une recherche approfondie. L'accessibilité de ces interfaces étant assez limitée et souvent réservée au milieu de recherche universitaire, il est rare que le choix de s'investir dans une pratique instrumentale soit spontané [2].

C'est donc en observant les instruments audionumériques et plus particulièrement les interfaces gestuelles déjà existantes que j'ai amorcé ma démarche en tant qu'interprète et créateur. Le choix d'un instrument mature, ergonomique et correspondant à ma vision ne s'étant pas réellement présenté, j'ai entrepris de réaliser l'interface gestuelle correspondant à ma sensibilité.

2.3. Choisir de créer

Actuellement, l'accessibilité aux technologies nous permet d'envisager une multitude de possibilités afin de réaliser l'instrument idéal, l'instrument à notre image et né d'une vision, d'un rêve. Cet instrument rêvé devra répondre à une pratique artistique, un besoin créatif, voir la vision d'une œuvre.

La création d'une interface gestuelle implique beaucoup d'étapes de développement avant d'arriver à une œuvre produite sur scène. Il était donc essentiel que j'établisse les critères qui me permettraient de me projeter dans l'avenir avec ma nouvelle interface gestuelle soit : l'ergonomie, la reproductibilité du geste musical et la transmission. Ces paramètres sont essentiels à une pratique instrumentale, à la performance sur scène et au développement de cette culture émergente.

[1] Il en sera peut-être autrement dans une quinzaine d'années.

3. LE GESTE DE L'INSTRUMENTISTE

Le geste instrumental est un sujet très étudié dans le cadre de la conception des nouvelles lutheries.[2] J'aborderai ce sujet avec le regard de l'interprète, celui du développement de l'expertise instrumentale ; ce qui donne à chaque interprète-instrumentiste son identité de performeur.[3]

3.1. Le geste expert et le geste inné

Pour Jean Geoffroy, percussionniste et interprète de l'œuvre *Light Music*[4] du compositeur Thierry De Mey, le geste est un tout qui appartient à tous. Tous les gestes proviennent de gestes « innés » [6] acquis par mimétisme. Ce geste « inné » devient la base d'un vocabulaire qui, travaillé par le musicien, deviendra le geste « expert » aussi nommé geste instrumental.

Le geste artistique est également crucial et donne tout son sens à la technique instrumentale qui, selon Jean Geoffroy, est la combinaison de l'« inné » et de l'« expert ». C'est l'équilibre de tous ces gestes qui révèle l'interprétation d'une œuvre. Un mélange d'instinct et d'intellect qui une fois sur scène et, ajouté à la personnalité du musicien proviendrait plus de l'instinct. Peu importe la quantité de travail et de répétition, les circonstances dans lesquelles une œuvre est interprétée font transparaître le geste « inné », le geste qui est propre à l'instrumentiste, son vocabulaire.

Ce vocabulaire est donc relatif à un instrument pratiqué pendant de nombreuses heures. Il est associé à un instrument en particulier (geste idiomatique) et à un individu particulier (geste idiosyncrasique). En me basant sur cette réflexion, j'ai étudié mes propres gestes afin de créer un système interactif qui me corresponde.

3.2. Le geste périphérique et corporéité

Jouer d'un instrument demande un engagement du corps tout entier pour produire le son. Le geste périphérique (ou ancillaire), qui accompagne la production du son, est bien souvent inconscient, comme le transfert de poids d'une jambe à une autre, mais il peut aussi être volontaire et permettre au spectateur de « voir le son » comme cela a été démontré avec le staccato du marimba[5] [8]. Dans les deux cas, le geste périphérique altère peu l'onde sonore elle-même mais peut influencer sur la perception. Il ajoute à la compréhension, à l'intention musicale.

La synthèse des gestes innés, experts et périphériques font l'identité de l'interprète, l'empreinte qu'il laisse dans une œuvre. On peut alors parler de corporéité [4] pour décrire cet ensemble de gestes.

Le corps dans sa globalité étant au centre de mes préoccupations musicales, j'ai identifié l'utilisation du geste dans l'espace et la perception du son par le visuel comme étant des éléments de ma corporéité.

Figure 1. Enchaînement du geste périphérique d'une marimbiste, transfert de poids.

3.3. Le geste-son dans l'espace

Le geste libre, sans repère haptique, ni contrainte matérielle, c'est la vision que j'ai du geste dans l'espace. Rendre visible l'invisible, rendre la matière sonore tangible et manipulable. L'espace sonore devient l'instrument.

Dans le domaine de la nouvelle lutherie, la relation geste-son est maintenant bien définie [3], mais reste tout de même un défi pour les interfaces gestuelles. Ajouter la notion d'espace au geste-son est ce qui révèle ma vision créatrice et ma corporéité. C'est avec ce regard, que j'ai réfléchi à l'interface qui me permettrait de faire transparaître le geste-son dans l'espace et qui pourrait se mouler, s'adapter aux corps.

J'ai donc fait le choix d'utiliser la captation du mouvement par caméra et développer un système interactif et ergonomique pour la performance. Les premiers plans ont été spécifiquement pensés pour la performance scénique, mais l'interface pourrait éventuellement être utilisée dans un autre cadre de création.

Figure 2. Trois gestes-sons dans l'espace, images de mouvements non consécutifs utilisées dans une miniature (*Plasticité,* 2012).

[2]Dans cet article, je ne fais pas référence aux recherches sur le geste, maintes fois citées, de François Delalande, Claude Cadoz et Marcello Wanderley. Les références présentées sont issues des pensées et de l'analyse intuitive du percussionniste émérite Jean Geoffroy sur son geste. [6]

[3]Ce terme est utilisé dans ce contexte pour désigner un instrumentiste qui maîtrise l'art de la scène.

[4]Pièce musicale pour un chef solo, projections et dispositif interactif.

[5]Le staccato au marimba est un geste volontaire du percussionniste pour faire voir le son au spectateur ; sans ce geste l'auditeur ne pourrait distinguer le staccato.

4. LA PRATIQUE INSTRUMENTALE

4.1. Assiduité et exigences physiques

Acquérir les compétences pour maîtriser un instrument de musique requiert de nombreuses années et une régularité dans les pratiques quotidiennes. L'interprète se dévoue à la tâche : « Un musicien professionnel fait subir à son organisme des violences anatomiques équivalentes à celles de grands sportifs. Leur quotidien à tous deux est fait d'entraînement, d'apprentissage, de répétitions et de pratique. »[6] [5]

Cette idée est bien ancrée dans le milieu des instrumentistes acoustiques (instruments de l'orchestre par exemple), mais est souvent oubliée en nouvelle lutherie. Ancrer un geste, un réflexe dans le corps demande du temps et c'est une notion qu'il faut valoriser dans le développement des interfaces gestuelles.

4.2. Valorisation de la pratique

Valoriser la pratique instrumentale d'une interface gestuelle est l'une des clés du développement de ces nouveaux instruments. Le problème que rencontre l'instrumentiste qui désire s'investir dans le développement d'un langage musical est souvent confronté au manque d'ergonomie, de transportabilité de l'outil et aux nombreuses connectiques qui doivent être faites avant d'obtenir un son. Toutes ces étapes à franchir pour l'instrumentiste avant de pouvoir s'exprimer avec l'interface sont des obstacles à la pratique instrumentale quotidienne et doivent être réduites au minimum, voir éliminées.

5. REFLEXIONS GENERALES SUR LE DEVELOPPEMENT D'UNE INTERFACE

5.1. Éthique

Afin de réaliser une interface gestuelle efficace, il est essentiel de travailler en équipe, agir en collégialité et valoriser la place et l'expertise de chacun. Cependant, la poussée artistique doit être le moteur de la réalisation et du développement. La mise en commun des ressources et des compétences complémentaires viennent ainsi enrichir le projet créatif.

L'œuvre *Light Music* du compositeur Thierry De Mey a été ainsi réalisée grâce à la collaboration de Jean Geoffroy (percussionniste) et Christophe LeBreton (ingénieur musical) : « Ayant chacun leur domaine d'expertise, l'équipe de création a eu une vision forte et complète de l'œuvre et a travaillé à la réalisation d'un but commun, *Light Music*. De nombreuses fois, l'importance de cette équipe a été mentionnée ; elle est primordiale pour le cheminement vers la première scénique de l'œuvre, mais également pour en assurer la pérennité. »[7] [7].

C'est avec ce souci de l'éthique que le projet SIC - Scène Interactive pour la Créativité - est réalisé en collaboration. Dès les prémisses du projet, les fonctions et les engagements de chacun des membres de l'équipe[8] ont été établis.

5.2. Ergonomie

L'ergonomie n'a peut-être pas la même définition pour un créateur que pour un ingénieur, mais c'est un aspect essentiel pour faciliter la pratique et la performance scénique. Ce que je considère comme étant des critères ergonomiques facilitant la pratique instrumentale (brièvement abordés au point 4.2.) sont : la transportabilité, la maniabilité et la liberté de mouvement (interface sans fil).

5.2.1. Transportabilité et maniabilité

Pour pratiquer un instrument, il faut pouvoir le faire quotidiennement dans un environnement propice ; dans un lieu qui n'est ni un laboratoire, ni une salle de concert. L'interface doit pouvoir s'installer de manière semi-permanente et ne pas nécessiter une longue période de calibration. Le système doit être fonctionnel en quelques clics de souris. De plus, l'instrumentiste doit pouvoir pratiquer sans avoir recours à l'aide d'un deuxième intervenant pour faire fonctionner ou enchaîner une œuvre. L'instrumentiste doit être autonome.

5.2.2. Sans fil

La liberté de mouvement est obtenue ici grâce à une connexion « sans fil », c'est-à-dire sans connexion physique entre l'ordinateur et l'interface se trouvant sur scène. De cette manière, l'ordinateur est mis hors de la vue du spectateur et n'influence ni la scénographie ni le comportement de l'instrumentiste.

5.3. Transmission et pérennité

La pérennité des œuvres qui seront produites avec notre instrument passera par le geste et le son, et surtout par la relation qui s'établit entre l'un et l'autre, le « geste-son », plus que par l'interface ou l'outil qui sont des éléments dont la durée de vie est variable. L'interprète a donc un rôle à jouer dans cette chaîne tout autant que le compositeur ou le créateur de l'interface. Il ne peut y avoir de transmission si les œuvres sont créées sans laisser de traces ; elle doit se faire avec une partition revisitée et adaptée aux nouvelles réalités des interfaces gestuelles. L'enseignement d'un interprète à un autre ou

[6]Debès, I., Schneider, M.P. et Malchaire, J. "Les troubles de santé des musiciens", médecine du travail & ergonomie Vol. XL No. 3, 2003, p.110.

[7]Héon-Morissette, B. "Rien dans les mains... *Light Music*", in Circuit Vol. 22 No. 1 La synchronisation, Les Presses de l'Université de Montréal, Montréal, Canada, p. 47.

[8]Patrick St-Denis et l'auteur de cet article, Barah Héon-Morissette sous la supervision du Professeur Jean Piché dans les laboratoires de l'IACT.

la pratique collective pourraient constituer des solutions à ces enjeux.

6. SIC - SCENE INTERACTIVE POUR LA CREATIVITE

SIC est un système de captation du mouvement constitué de deux parties indépendantes. La première composante utilise la reconnaissance de geste via le périphérique Kinect alors que la seconde est constituée d'un sol sensible. Les deux composantes peuvent être utilisées ensemble ou séparément dans le cadre d'une performance scénique ou dans d'autres contextes nécessitant un système interactif.

6.1. kinKI - Kinect Kreative Interface

kinKi, développée dans les laboratoires de l'IACT en collaboration avec Patrick St-Denis, est une application de détection du mouvement créée avec le logiciel OpenFrameworks et utilisant le capteur Kinect conçu pour la console de jeux vidéo Xbox 360. L'application kinKI constitue le premier module de SIC.

Après plusieurs essais peu concluants avec d'autres technologies et librairies d'objets Max/MSP/Jitter tels des caméras vidéo, des caméras Web FireWire, les objets softVNS[9], Jamoma[10], cv.jit[11], l'utilisation de la Kinect est devenue évidente. Ce périphérique associé à un langage de programmation solide comme OpenFrameworks est très performant, ne présente pas de latence vraiment perceptible, est ergonomique et peu coûteux.

6.1.1. kinKI 0.0.2

Cette première version permet la reconnaissance de treize points du squelette. Chacun de ces points assignés à des zones circulaires agit comme déclencheur. Ces zones sont enregistrées lors d'un processus de captation de pose fait par l'utilisateur. La tolérance de chaque zone peut être déterminée et peut varier pour chacune. Les valeurs obtenues par les points du squelette sont associées à l'axe des x et y. Il est aussi possible de relier deux zones de déclenchement par une ligne et ainsi faire varier les valeurs de l'une à l'autre.

En assignant un numéro d'identification, huit squelettes peuvent être reconnus simultanément. L'interface de l'utilisateur permet également d'enregistrer jusqu'à cent zones par scène et jusqu'à cent scènes par programme. Les zones et les scènes peuvent être sauvegardées en fichier.

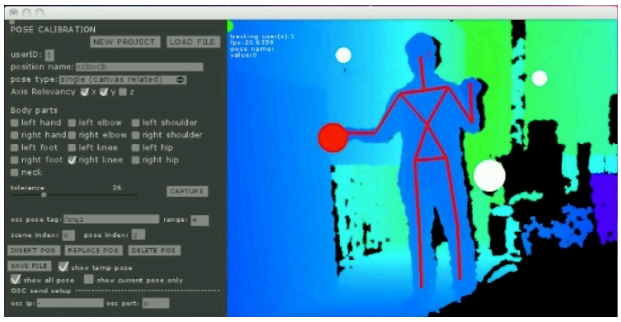


Figure 3. Interface-utilisateur de kinKI 0.0.2.

6.1.2. kinKI 0.0.3

kinKI 0.0.3 est une version 3D. Contrairement à la version précédente, l'axe z est également utilisé. L'environnement entourant le squelette a été éliminé pour être remplacé par un cube virtuel de couleur jaune sur fond noir représentant l'espace de scène. Les zones représentées par des points sont maintenant déterminées par des cubes identifiés comme des objets. Ces objets passent du rouge au vert lorsqu'ils sont activés par le point du squelette auquel ils ont été assignés. À l'intérieur, le point central de l'objet sert de centre de gravité alors qu'un axe le relie au point du squelette qui entre dans la zone cubique. Lorsque l'utilisateur entre dans l'objet, la valeur « 1 » est d'abord envoyée puis six valeurs correspondant à l'axe des x, y et z entre « 0. et 1. » ainsi que trois valeurs d'angles.

En mode édition, l'utilisateur détermine l'emplacement et la dimension de chacun des objets. Ils peuvent être superposés et imbriqués. L'élément principal qui diffère de la version 0.0.2 est la possibilité d'avoir des objets fixés dans l'espace de jeu alors que d'autres objets bougent avec le squelette en mode *body related*.

Une représentation de l'espace sur 4 plans a été ajoutée pour faciliter le repérage des objets par l'interprète : plan de face, de côté, d'en haut et une vision du narrateur.

L'interprète peut manipuler l'interface seul, sans l'aide d'un autre individu, en chargeant un fichier préalablement sauvegardé. En mode concert, l'interprète peut passer d'une scène à l'autre pour changer la disposition des objets dans l'espace.

kinKI 0.0.3 est beaucoup plus complexe que la première version et a décuplé les possibilités musicales et gestuelles. La complexité du système demande une plus grande maîtrise de l'interface pour arriver à un geste expert.

[9]Librairie d'analyse du mouvement et traitement de la vidéo en temps réel pour Max/MSP/Jitter développé par David Rokeby, http://homepage.mac.com/davidrokeby/softVNS.html

[10]Modules d'analyse du mouvement développé par une équipe de collaborateurs, www.jamoma.org

[11]Librairie d'analyse d'image pour Max/MSP/Jitter développé par Jean-Marc Pelletier, www.jmpelletier.com

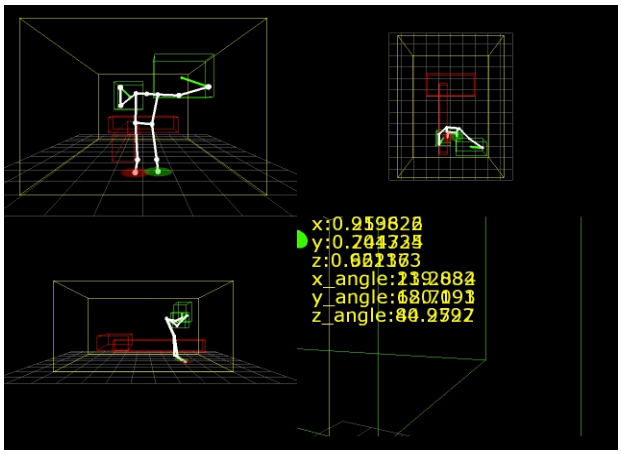


Figure 4. Représentation de l'espace en 4 plans de kinKI 0.0.3.

6.2. Sol sensible

Cette composante de la «Scène interactive pour la Créativité » (SIC) est constituée d'une grande dalle captant le transfert de poids de l'instrumentiste grâce à des capteurs de force Interlink[12] placés sous un panneau de fibres à densité moyenne (MDF).

L'amplitude du geste de l'interprète étant variable, nous avons conçu ce module en deux formats différents pour mieux correspondre au geste périphérique de divers instrumentistes. Naturellement, le clarinettiste ne se déplace pas autant dans l'espace que le ferait un percussionniste ou encore un danseur. C'est dans cette optique que j'ai construit une version de 60 cm x 60 cm et une autre de 120 cm x 120 cm[13].

Actuellement, les capteurs sont branchés dans un microcontrôleur Wi-microDig[14]. Ce périphérique Bluetooth à 8 entrées analogiques est très simple d'utilisation et excellent pour le prototypage. Éventuellement, nous utiliserons un microcontrôleur plus abordable et sans fil.

Les valeurs des capteurs générées par le poids du corps sur la surface peuvent être utilisées indépendamment les uns des autres ou alors ensemble pour déterminer la position du poids sur la surface dans les axes x et y.

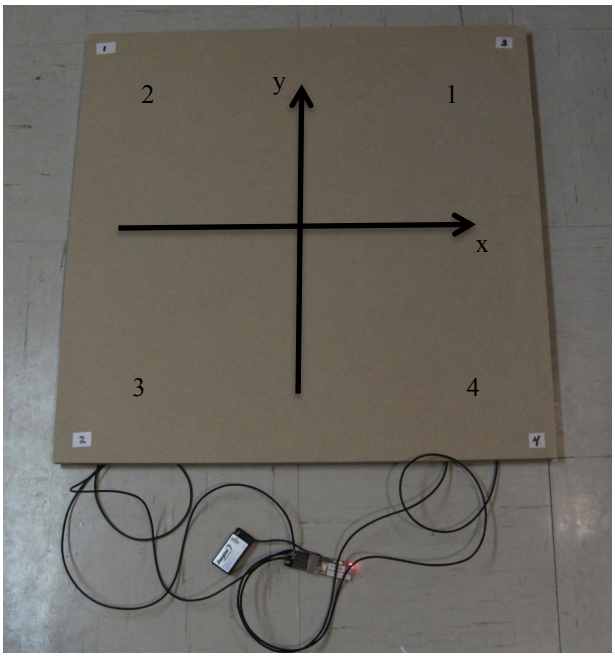


Figure 4. Sol sensible 60 cm x 60 cm à quatre capteurs.

6.3. Le son de SIC

SIC est à ce jour encore en développement. Une première œuvre est prévue pour l'automne 2012. L'aspect sonore reste encore à explorer ainsi que la maîtrise du geste expert pour cette interface gestuelle. Pour l'instant, nous prévoyons développer notre stratégie de mapping via le logiciel Max/MSP. Le mapping étant pour nous une étape relative à la composition d'une œuvre, nous n'en exposerons pas les détails dans cet article. L'interface comme l'instrument est un outil au service de la création.

7. CONCLUSION

SIC a été conçu pour répondre à mon besoin de créativité et d'expressivité, mais il s'avère que beaucoup d'avenues sont possibles pour cette interface gestuelle. Les deux composantes, kinKI et le sol sensible, ont des perspectives créatives dans plusieurs domaines comme la danse, les installations sonores et la pédagogie. De plus, les qualités ergonomiques de SIC font de cette interface gestuelle un outil créatif accessible pour le développement d'une pratique collective.

8. REMERCIEMENTS

Je tiens à remercier Prof. Jean Piché et Prof. Caroline Traube et, mon collaborateur Patrick St-Denis, l'Institut Arts Cultures et Technologies (IACT, le Fonds de recherche sur la société et la culture (FQRSC) pour son soutien financier et l'Observatoire interdisciplinaire de création et de recherche en musique (OICRM) pour la bourse de déplacement.

[12]Capteur standard 406 FSR, http://www.interlinkelectronics.com/Product/Standard-406-FSR

[13]Valeurs arrondies lors de la conversion du système impérial au système métrique.

[14]Périphérique commercialisé par Infusion Systems, www.infusionsystems.com

9. REFERENCES

[1] Bruser, M. "The Art of Practicing, A Guide to Making Music from the Heart", Bell Towe, New York, 1997.

[2] Cance, C. et Genevois, H. "Questionner la notion d'instrument en informatique musicale : analyse des discours sur les pratiques du méta-instrument et de la méta-mallette", Actes des 14èmes Journées d'Informatique Musicale, ACROE et laboratoire ICA, Grenoble, 2009, p. 133-142.

[3] Caramiaux, B. "« Gestification » du son : mapping adaptatif geste/son dans un contexte d'écoute et de performance musicale", Master 2 Recherche, Mention Informatique, spécialité SAR ; en collaboration avec ParisTech et l'IRCAM - Centre Pompidou). Université Pierre et Marie Curie, Paris, Franec, 2008.

[4] Caullier, J. "La corporéité de l'interprète", in L'imaginaire musicale entre création et interprétation, L'Harmattan, Paris, France, 2006, p.133-150.

[5] Debès, I., Schneider, M.P. et Malchaire, J. "Les troubles de santé des musiciens", médecine du travail & ergonomie Vol. XL No. 3, 2003.

[6] Geoffroy, J. "Le geste dans l'œuvre musicale, la musique et le mouvement", in Rencontres musicales pluridisciplinaires : Le Feedback dans la création musicale, GRAME, Lyon, 2006.

[7] Héon-Morissette, B. "Rien dans les mains... *Light Music*", in Circuit Vol. 22 No. 1 La synchronisation, Les Presses de l'Université de Montréal, Montréal, Canada, p. 41-50.

[8] Schutz, M. "Seeing Music? What musicians need to know about vision" in Empirical Musicology Review Vol. 3 No. 3, Ohio State University Library, Ohio, 2008.

LOOPJAM: UNE CARTE MUSICALE COLLABORATIVE SUR LA PISTE DE DANSE

Christian Frisson, Stéphane Dupont, Julien Leroy, Alexis Moinet, Thierry Ravet, Xavier Siebert, Thierry Dutoit
Université de Mons (UMONS), laboratoire TCTS et MathRO
Boulevard Dolez 31 B-7000 Mons, Belgium
Courriels: *prenom.nom*@umons.ac.be

RÉSUMÉ

Ce papier présente l'installation LoopJam qui permet aux visiteurs d'interagir avec une carte musicale par le biais d'un système de suivi gestuel par vision informatique. La carte sonore résulte d'un partitionnement des sons en groupes par similarité basée sur leur signal. Le rendu sonore est contrôlé par les positions ou gestes des participants captés par une caméra Kinect détectant la profondeur de la scène 3D. Les mouvements des participants exprimant une mesure ou un tempo sont corrélés à la vitesse de lecture commune à tous les échantillons synchronisés par le moteur audio. Nous avons présenté et testé une première version de cette installation lors de trois expositions en Belgique, Italie et France. Les réactions parmi les participants ont varié entre la curiosité et l'amusement.

1. INTRODUCTION

L'intention de l'installation LoopJam est d'inciter ses visiteurs à construire collaborativement une atmosphère musicale en direct. Une carte sonore bidimensionnelle est constituée automatique par notre logiciel qui analyse les échantillons sonores et les regroupe par similarité. En parcourant l'installation, chaque visiteur explore cette carte sonore et déclenche des évènements sonores par des gestes simples qui déterminent également le tempo. La lecture de chaque son est synchronisé par notre logiciel, permettant ainsi une composition collaborative musicale en direct créée par tous les visiteurs simultanés.

Dans la partie 2 nous dégageons le contexte de cette installation et décrivons brièvement les travaux connexes. Dans la partie 3 nous détaillons l'architecture logicielle et les choix d'implémentation. Dans la partie 4 nous discutons des conclusions après avoir observé des visiteurs utilisant l'installation. Dans la partie 5 nous proposons de potentiels améliorations que nous envisageons pour cette installation.

2. TRAVAUX SIMILAIRES

2.1. Domaines d'application

L'objectif principal de l'installation LoopJam est de permettre aux visiteurs de créer un rendu musical interactif et collaboratif. Les ancêtres informatisés les plus directs à cette installation depuis les années 1950 sont les environnements de synthèse sonore, donnant accès à des paramètres d'algorithmes de traitement de signal audio, et les séquenceurs audio multipistes, inspirés directement des techniques tangibles de découpage et recollage de bandes magnétiques. Ces systèmes ont proposé des interfaces utilisateur à convivialité variable, à l'origine à l'aide de cartes trouées ou de terminaux par saisie de lignes de commandes, en temps différé selon les limitations de traitement informatique ; par la suite des interfaces graphiques associées à des souris, claviers et autres contrôleurs munis de potentiomètres rotatifs ou linéaires, etc... fournissant une interactivité plus attirante, en temps-réel. Récemment un intérêt fort a été porté sur les interfaces utilisateur tangibles and les dispositifs multi-point, dans l'intention de regagner une interaction gestuelle avec la musique, pourtant les interfaces "WIMP" (de l'anglais : Fenêtre Icône Menu Pointeur) sont toujours majoritairement employées dans les studios des professionnels de la musique.

L'arrivée des ordinateurs personnels et des tablettes a popularisé l'utilisation de ces systèmes, des experts et chercheurs vers les amateurs. Une convergence entre les milieux des performances musicales et de l'industrie du jeu vidéo, particulièrement les séries *Guitar Hero* et *Rock Band*, a suscité un intérêt pour la création musicale chez davantage d'utilisateurs pas nécessairement virtuoses mais désirant allier créativité et simplicité.

2.2. Une taxonomie

LoopJam combine un ensemble des technologies éprouvées et émergentes : le traitement de signal audio et musical, la fouille dans les données musicales, la navigation audio, la vision informatisée 3D, le contrôle gestuel, la reconnaissance d'expressions corporelles.

Les travaux connexes peuvent être classés selon les caractéristiques suivantes :

- Nature du projet : est-ce une installation physique, une application spécifique à une plateforme ou navigable en ligne ?
- Comment est produit le rendu sonore ? Par le biais de synthèse sonore ou d'échantillonnage ?
- Quelle est la frontière entre la composition et la performance musicales déterminée par le système ? Le rendu sonore est-il en temps réel ou différé ?

– Quelles modalités d'interaction sont proposées ? Des capteurs portés ou manipulés sont-ils nécessaires ? Ou une interaction moins invasive et à distance est-elle choisie ?
– Combien d'utilisateurs peuvent interagir simultanément avec le système ? Un seul ? Plusieurs ?
– Y-a-t-il plusieurs types d'utilisateurs impliqués à différents niveaux d'interaction avec ces systèmes : les utilisateurs directs, le public, des DJs ou "maîtres de cérémonie" (MC's)... ?

2.3. Quelques exemples de systèmes multi-utilisateurs de composition musicale interactive

Suivant la taxonomie dégagée juste avant, nous allons caractériser particulièrement les travaux proposant des installations (si possible multi-utilisateurs) qui permettent de produire une composition sonore (si possible musicale) assistée et basée sur échantillons sonores.

Dance Jockey permet la manipulation en temps réel de musique selon les mouvements corporels de danseurs [4]. Des capteurs portés sont requis pour l'analyse des mouvements, ce qui peut limiter le nombre d'utilisateurs simultanés et par leur nature invasive réduire la spontanéité lors de la phase de découverte de l'installation (les visiteurs devant au préalable "s'habiller" de costumes de capteurs avant de pouvoir interagir).

SoundTorch propose de naviguer dans une carte sonore à l'aide d'une télécommande Nintendo Wii similairement à la découverte progressive d'un espace réel sombre à l'aide d'une lampe torche [9]. Ce projet est initialement une application pour ordinateur personnel, mais pourrait devenir une installation de navigation collective en augmentant le nombre de télécommandes comme prévu pour la console de jeux pour laquelle ces télécommandes ont été conçues.

BeatScape [1] fournit une interface utilisateur plus complexe : un premier groupe de performeurs utilise une interface tangible sur table qui permet de positionner des échantillons sonores qui seront déclenchés par un second groupe de performeurs utilisant des télécommandes Wii, ce qui permet une composition collaborative à divers niveaux d'assignements de tâches.

LoopJam se différencie de ces travaux en proposant une interface utilisateur non invasive grâce à une caméra Kinect et en fournissant une organisation automatique des échantillons sonores par similarité de timbre (et par incidence majoritairement par instrument) suite à l'analyse par traitement de signal de leur contenu.

3. IMPLÉMENTATION

3.1. Architecture

La figure 1 donne une vue d'ensemble de la configuration de l'installation : une carte sonore 2D est projetée sur un mur vertical, plusieurs visiteurs ou joueurs se déplacent sur le sol et leur position correspond à des curseurs de navigation sur la carte affichée.

Figure 1. Photographie de l'installation avec 3 participants.

La figure 2 détaille l'architecture : 4 haut-parleurs sont disposés autour de la zone de captation et fournissent un son spatialisé, les participants sont captés à l'aide d'une caméra Kinect.

3.2. MediaCycle, une suite logicielle pour l'organisation de contenu multimedia par similarité

Nous développons depuis trois ans MediaCycle, une suite logicielle pour l'organisation par similarité de contenu multimedia. Les types de média supportés actuellement sont : audio (des échantillons aux pistes musicales [5]), images, video (particulièrement pour une installation pour explorer des vidéos de danseurs [6]) et texte (en tant que métadonnée pour d'autres types de média ou type à part entière). La suite logicielle propose des implémentations compatibles autant pour des ordinateurs isolés que des applications client/serveur, avec un but commun de fournir une représentation multi-dimensionnelle, visuelle et abstraite, d'une "médiathèque". Son architecture est modularisée par type de média ; par algorithmes d'extraction de caractéristiques dédiées, de classification (l'algorithme standard "k-moyennes" est principalement utilisé), et de calcul de positions de noeuds-media dans une représentation réduite à deux dimensions.

LoopJam et une application multi-plateforme créée avec la suite logicielle MediaCycle [5]. Les fichiers sonores représentés par des noeuds visuels peuvent être parcourus rapidement avec un retour sonore instantané.

3.2.1. Auto-organisée par : similarité de timbre, instrument

Pour organiser les audiothèques visuellement selon des caractéristiques haut-niveau comme le timbre, la rythmique et l'harmonie ; nous avons implémenté plusieurs algorithmes de caractéristiques bas-niveaul : spectrales (centroïde, entropie, variation, MFCC ou coefficients cepstraux sur l'échelle de Mel, taux de passage au niveau moyen), perceptuels (intensité perçue), temporels (temps d'attaque logarithmique, fréquence et amplitudes de modulation d'énergie).

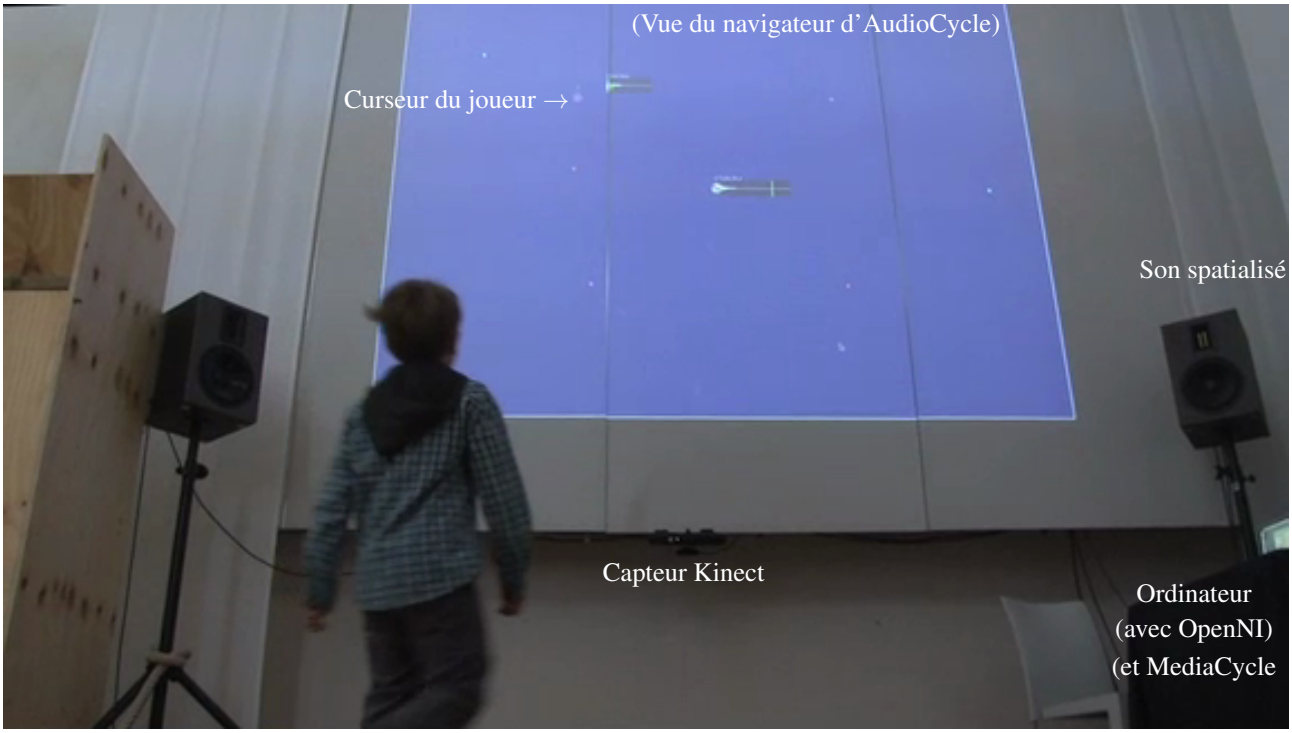


Figure 2. Photographie de l'installation annotée avec des descripteurs de son architecture.

En favorisant un choix de boucles mono-instrumentales, l'analyse du timbre permet donc de classer les sons par "instrument".

3.2.2. Synchronisée sur un tempo commun

Les collections d'échantillons sonores usuellement importées dans l'installation LoopJam utilisent le format ACID, pour lesquelles le tempo est encodé en tant que métadonnée, même si d'autres formats audio sont supportés. Dans le cas du format ACID, LoopJam synchronise automatiquement la vitesse de lecture des échantillons à un tempo commun à l'aide d'un vocodeur de phase.

3.2.3. Spatialisé dans le plan horizontal

Le moteur audio créé pour MediaCycle utilise la librairie OpenAL pour le rendu sonore spatialisé. Les sons sont spatialisés dans le plan horizontal, par défaut la configuration standard est une simple stéréophonie, mais compatible aussi avec une configuration quadriphonique utilisée pour LoopJam pour une meilleure immersion tout en gardant l'installation transportable.

3.3. Une interaction dite naturelle, par les gestes

Nous avons implémenté un contrôle réseau de la suite logicielle MediaCycle par le protocole OpenSoundControl (OSC) initialement pour utiliser des contrôleurs standard pour la navigation inter- et intra-media : molettes rotatives, souris 3D, joysticks à retour d'effort... [7]. Nous avons mis à jour cet interfaçage pour supporter plusieurs curseurs de navigation et ainsi permettre à plusieurs utilisateurs d'interagir simultanément avec la carte sonore. Nous avons utilisé une caméra Kinect qui capte la profondeur couplé à la librairie OpenNI pour segmenter la scène visuelle 3D d'utilisateurs.

Une option de l'installation LoopJam est de modifier le tempo des échantillons sonores par l'analyse des mouvements des utilisateurs. Nous avons implémenté un algorithme qui mesure la rythmique que les utilisateurs battent. La vitesse de la main gauche est extraite de la représentation du squelette des participants. Nous calculons la somme de ces vitesses et extrayons le maximum de puissance de l'analyse spectrale du signal sur une fenêtre de 2 secondes. La fréquence obtenue est moyennée par un filtre passe bas, cette valeur obtenue contrôlant le vocodeur de phase implémenté dans MediaCycle.

L'utilisateur faisant les mouvements les plus amples influence le tempo majoritairement.

3.4. Collections de boucles audio utilisées dans LoopJam

Parmi les collections de boucles audio ayant été utilisées dans LoopJam :

- des librairies de boucles au format ACID, commerciales mais libres de droit pour la retransmission, par Zero-G (par exemple le DVD *Pro Pack*) et Sony Creative Software (par exemple *8-bit Weapon : A Chiptune Odyssey*) ;
- des collections libres telles que la libraire de boucles libres One Laptop Per Child (OLPC).

4. DISCUSSION

Cette première version de notre installation a été exposée à Seneffe en Belgique pour l'évènement Arts & Sciences les 21 et 22 mai 2011. Elle a également été installée à Paris au Centre Wallonie-Bruxelles, du 22 septembre au 23 octobre 2011. LoopJam a été sélectionné pour le concours artistique du Network & Electronic Media (NEM) Summit à Turin, Italie, les 25-27 septembre 2011, et classé par le jury dans le top 5.

Nous discutons ci-après des choix initiaux après un examen et interrogation des visiteur lors de la première exposition.

4.1. L'organisation par similarité aide-t-elle ?

L'organisation par similarité basée sur le contenu est essentielle pour cette installation pour deux raisons. Premièrement, comme nous avons choisi de classer les sons par similarité de timbre, le chemin parcouru par chaque joueur dans l'installation et par incidence dans la représentation des sons est consistante ; les participants s'attendent à trouver des sons du même instrument dans une même zone, à la manière de la localisation des instrumentistes dans un orchestre classés par type d'instrument (sections de cordes, de cuivres, etc...). Deuxièmement, l'organisation automatique facilite la tâche de préparer le contenu de chaque collection de sons parcourue dans LoopJam, en opposition à un classement totalement manuel.

4.2. Interaction directe ou indirecte ?

Nous avions initialement prévu de projeter la carte sonore sur le sol pour permettre une interaction directe [3], les participants auraient alors vu la représentation visuelle à leurs pieds, en correspondance spatiale directe avec la représentation de la zone d'interaction. Cependant ceci aurait impliqué un certain nombre d'inconvénients en comparaison avec la configuration actuelle : le fait que les gens orientent leur regard au sol incite moins au contact social, des noeuds visuels pourraient être masqué par occlusion avec des participants, une configuration robuste vis-à-vis de l'occlusion aurait requis du matériel bien moins transportable (par exemple une plateforme horizontale avec rétroprojection).

4.3. Quelle est l'expressivité des visiteurs, préparateurs et responsables de l'installation ?

A travers cette installation, nous redéfinissons la relation entre les visiteurs et les préparateurs de l'installation. L'"audience" peut participer à la performance musicale en dansant, son comportement et ses mouvements affectant le choix des échantillons sonores ainsi que les paramètres de lecture (tempo, volume). Le préparateur de l'installation peut insuffler sa sensibilité artistique à travers le choix des collections sonores présentées [10], d'autant plus grande mesure quand les sons sont créés par le préparateur-même. Les DJs faisant attention aux signes de désengagement du public [8], le responsable de l'installation peut également influencer sur l'ambiance générale pendant la performance en variant les paramètres de l'organisation des collections : choix de collections, navigation plus précise dans une collection par zoom ou rotation spatiale. S'il s'agit de la même personne, le préparateur et responsable de l'installation peut donc assurer une certaine qualité de la performance musicale, avant et pendant, laissant s'exprimer les participants en leur laissant une variabilité contrôlée et assistée des paramètres à leur disposition.

4.4. Comment maintenir la satisfaction des visiteurs ?

Notre méthode actuelle pour garder les participants satisfaits et apprécier l'installation est majoritairement de varier fréquemment la collection musicale exposée pendant la durée de l'installation, ce qui est automatisé avec un séquencement. Nous avons choisi de ne pas ré-organiser une collection pendant une performance car nous pensons que les utilisateurs se constituent une carte mentale des sons similaire à leur représentation visuelle. Comme les visiteurs consacrent un temps assez réduit à l'installation, ils pourraient être découragés et perturbés par les changements de représentations, à la manière d'un instrumentiste auquel on imposerait un accordage non-standard en performance. Les méthodes de navigation telles que le zoom et la rotation permettent un juste milieu : affiner la vue tout en conservant le même modèle. Les rotations permettent également aux visiteurs d'intervertir les instruments.

5. CONCLUSIONS, PERSPECTIVES

Nous avons proposé une installation qui mêle informatique musicale et interaction multi-utilisateurs pour composer et "performer" de la musique en s'amusant. L'auto-organisation par similarité de timbre basée sur le contenu accélère résolument l'import de collections de boucles sonores pour la préparation ou personnalisation de l'installation : et fournit une représentation spatiale des boucles sonores organisées par instrument à la manière des instrumentistes d'un orchestre.

Testée au cours de trois expositions, cette installation pourrait être transposée à d'autres types de lieux comme les boîtes de nuit, les institution socio-culturelles, des parcs d'attraction...

Le nombre de boucles dans chaque collection, l'espace résultant entre chaque noeud représentant une boucle, et la taille de ces noeuds pourraient rendre l'installation trop sensible aux déplacements de faible amplitude des visiteurs. Des possibilités pour compenser cet inconvénient seraient d'adapter la taille des collections et la dimension de l'espace sensible en utilisant plusieurs caméras combinées et fusionnées.

Si nous pouvons déjà extraire des représentations du squelette de chaque participant dans la scène 3D grâce à la caméra, nous pourrions envisager de parfaire la détection de rythmique individuellement afin de permettre le

changement de tempo par boucle.

Nous pensons à d'autres façons de personnaliser l'installation. Les téléphones mobiles des participants pourraient être utilisés pour synchroniser avec leurs profils sur des sites de recommandation tels que LastFM et récupérer l'historique de leurs goûts musicaux, alors que les téléphones de l'audience pourraient servir à voter sur le choix des collections de sons, ou proposer aux joueurs de "jammer" ensemble, voire de céder leur place ; éventuellement par manifestation vocale [2]. Nous pourrions remplacer les curseurs visuels de chaque participant par des avatars générés grâce à la caméra.

Enfin, à l'aide d'analyse informatisée des mécanismes d'attention sociale, nous pourrions améliorer la jouabilité de l'installation : donner plus de poids aux participants aux comportements saillants, inviter d'autres gens parmi l'audience à participer.

6. REMERCIEMENTS

`numediart` est un programme de recherche autour des technologies des arts numériqus, financé par la Région Wallonne, Belgique (grant N°716631).

Nous voudrions remercier notre ancien collègue Damien Tardieu pour avoir suggéré le concept de cette installation.

Notre travail aurait moins de visibilité sans les démos vidéo professionnelles de Laura Colmenares Guerra.

7. REFERENCES

[1] A. Albin, S. Senturk, A. V. Troyer, B. Blosser, O. Jan, and G. Weinberg. Beatscape and a mixed virtual-physical environment for musical ensembles. In A. R. Jensenius, A. Tveit, R. I. Godøy, and D. Overholt, editors, *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 112–115, Oslo, Norway, 2011.

[2] L. Barkhuus and T. Jørgensen. Engaging the crowd : studies of audience-performer interaction. In *CHI '08 extended abstracts on Human factors in computing systems*, CHI EA '08, pages 2925–2930, New York, NY, USA, 2008. ACM.

[3] S. K. Card, J. D. Mackinlay, and G. G. Robertson. The design space of input devices. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI'90)*, pages 117–124, 1990.

[4] Y. de Quay, S. A. v. D. Skogstad, and A. R. Jensenius. Dance jockey : Performing electronic music by dancing. *Leonardo Music Journal*, 21 :11–12, 2011.

[5] S. Dupont, C. Frisson, X. Siebert, and D. Tardieu. Browsing sound and music libraries by similarity. In *128th Audio Engineering Society (AES) Convention*, London, UK, May 22-25 2010.

[6] C. Frisson, S. Dupont, X. Siebert, and T. Dutoit. Similarity in media content : digital art perspectives. In *Proceedings of the 17th International Symposium on Electronic Art (ISEA 2011)*, Istanbul, Turkey, September 14-21 2011.

[7] C. Frisson, S. Dupont, X. Siebert, D. Tardieu, T. Dutoit, and B. Macq. DeviceCycle : rapid and reusable prototyping of gestural interfaces, applied to audio browsing by similarity. In *Proceedings of the New Interfaces for Musical Expression++ (NIME++)*, Sydney, Australia, June 15-18 2010.

[8] C. Gates, S. Subramanian, and C. Gutwin. Djs' perspectives on interaction and awareness in nightclubs. In *Proceedings of the 6th conference on Designing Interactive systems*, DIS '06, pages 70–79, New York, NY, USA, 2006. ACM.

[9] S. Heise, M. Hlatky, and J. Loviscach. Soundtorch : Quick browsing in large audio collections. In *125th Audio Engineering Society Convention*, number 7544, 2008.

[10] T. Rodgers. On the process and aesthetics of sampling in electronic music production. *Organised Sound*, 8(3) :313–320, December 2003.

IANNIX 0.8

Guillaume Jacquemin
Association IanniX
guillaume@iannix.org

Thierry Coduys
Association IanniX
thierry@iannix.org

Matthieu Ranc
Association IanniX
matthieu@iannix.org

RESUME

IanniX est un séquenceur graphique open source, inspiré des travaux de Iannis Xenakis et destiné à la création numérique [7]. Le logiciel propose une écriture polytemporelle du temps d'événements statiques et dynamiques vers un environnement dédié (Processing, PureData, SuperCollider…).

À l'aide d'une palette d'objets fondamentaux que sont les *triggers (événements)*, les courbes *(trajectoires dans l'espace)* et les curseurs *(progression dans le temps)*, IanniX permet une représentation graphique et interactive du temps dans l'espace 3D et assure un échange bidirectionnel via plusieurs protocoles de communication, dont l'OSC[1].

Depuis mai 2011, l'architecture générale de IanniX a été réétudiée et de nouvelles fonctionnalités innovantes enrichissent ses possibilités : scripts génératifs, écriture récursive, live coding, nouvelles interfaces réseau, gestion intégrale de la 3D. Aussi, pour assurer un financement potentiel (subventions, dons) et garantir les développements futurs, l'association loi 1901 « IanniX » a été créée[2].

Nous présentons ici un état des lieux de IanniX, de ses intentions initiales aux perspectives ouvertes par les nouveaux développements.

1. INTENTIONS

1.1. De l'UPIC à IanniX

Dans les années 1970, Iannis Xenakis entreprenait la réalisation de l'UPIC[3]. Cet outil, extension de la notation du solfège traditionnelle, permettait au musicien de composer dans un espace temps / fréquence et de l'interpréter de manière simple et directe, en conservant une notion de geste instrumental.

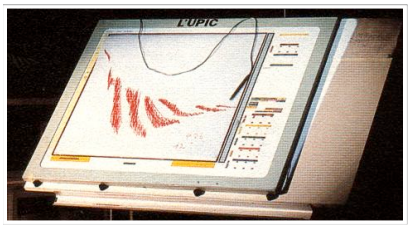


Figure 1. Partition dessinée sur l'UPIC

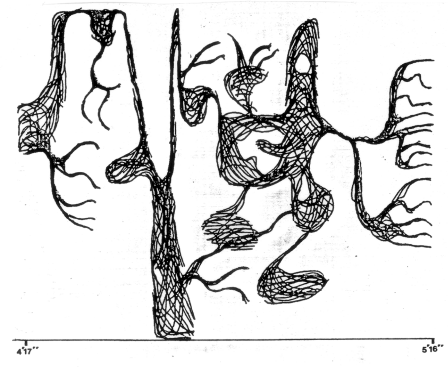


Figure 2. Partition graphique de Iannis Xenakis, Mycenæ Alpha, composée sur UPIC

Les développements successifs qu'a inspiré l'UPIC *(Metasynth, HighC…)* ont reproduit un outil de dessin spectral, avec une synthèse de son intégrée — donc imposée —, sans introduire la notion de geste instrumental ni de temps réel.

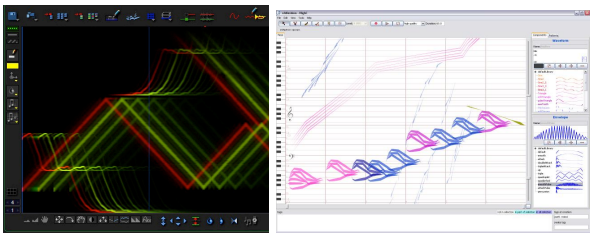


Figure 3. MetaSynth et High-C

1.2. Séquenceurs Open Sound Control

L'intégration progressive de l'Open Sound Control [6] dans les outils de création numérique a ouvert le champ à la création de nombreux séquenceurs OSC. Cependant, ces séquenceurs sont des adaptations de séquenceurs traditionnels et n'intègrent pas le geste instrumental ni de technique d'écriture / composition.

Seuls quelques séquenceurs (TimelinerSA [1], Open Timeline [3], la plateforme VIRAGE [4], AlgoScore [5]) introduisent une gestion graphique du temps avec une seule timeline en une dimension.

[1] Open Sound Control : protocole UDP pour le contrôle temps réel
[2] Association IanniX, 6/8 rue Notre-Dame de Nazareth – 75003 Paris — Statuts : http://iannix.org/ressources/statuts.pdf
[3] Unité Polyagogique Informatique du CEMAMu

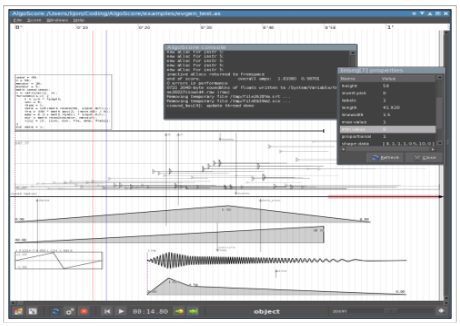


Figure 4. Capture d'écran d'AlgoScore

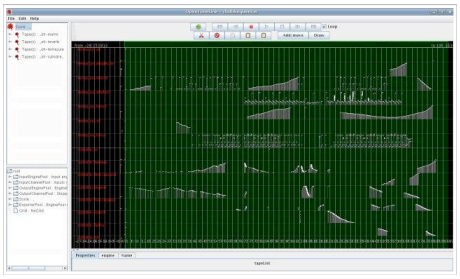


Figure 5. Capture d'écran d'OpenTimeline

1.3. Principe de IanniX

Sur les bases de l'UPIC précédemment décrites et dans une démarche innovante de séquencement Open Sound Control, IanniX a été imaginé comme un séquenceur graphique temps réel, permettant de construire et de composer des partitions graphiques. Une forte volonté d'abstraction a été imposée afin que IanniX ne soit pas dédié uniquement à l'informatique musicale, mais aussi au traitement du signal vidéo, au show-control pour les installations, et à la gestion de différentes interfaces hardware…

Ainsi IanniX s'interface naturellement avec l'environnement temps réel habituel du compositeur / créateur (PureData, CSound, Processing, SuperCollider,).

2. LES OBJETS FONDAMENTAUX

Pour composer une partition graphique temps réel dans IanniX, une palette restreinte de trois objets fondamentaux et distincts a été imaginée :

- les *triggers* déclenchent des événements ponctuels ;
- les *courbes* sont des suites de points dans l'espace tridimensionnel ;
- les *curseurs* évoluent sur des courbes et progressent en fonction du temps.

2.1. Événements : triggers

Un *trigger* T se définit littéralement par un point P dans l'espace 3D de coordonnées (x, y, z) :

$$T = P(x, y, z) \qquad (1)$$

La faculté d'un *trigger* est d'émettre un message lorsqu'il est déclenché, au passage d'un curseur.

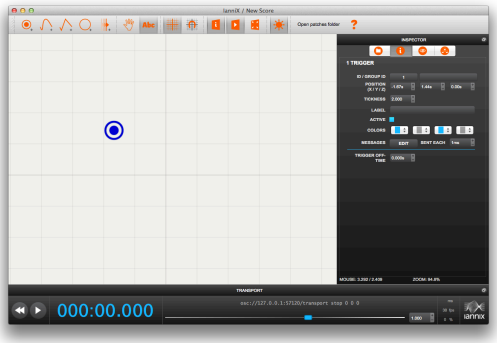


Figure 6. Un trigger dans IanniX

2.2. Trajectoires : les courbes

Une courbe S_n est définie en fonction du temps par une succession de n sections de courbes S_k associées à leur longueur L_k :

$$S_n(t) = \begin{cases} S_0\left(\frac{t}{L_0}\right), & t \in [0, L_0[\\ S_1\left(\frac{t-L_0}{L_1-L_0}\right), & t \in [L_0, L_1[\\ S_2\left(\frac{t-L_1}{L_2-L_1}\right), & t \in [L_1, L_2[\\ \vdots & \\ S_n\left(\frac{t-L_{n-1}}{L_n-L_{n-1}}\right), & t \in [L_{n-1}, L_n] \end{cases} \qquad (2)$$

Chaque section de courbe S_k peut prendre 3 formes : un segment de droite, une ellipse ou une courbe de Bézier cubique.

2.2.1. Segment de droites

Une section de courbe en segment de droite est définie en fonction du temps par deux points $P_{k,0}$ et $P_{k,1}$ telle que :

$$S_k(t) = P_{k,0} \cdot (1 - t) + P_{k,1} \cdot t, \; t \in [0, 1] \qquad (3)$$

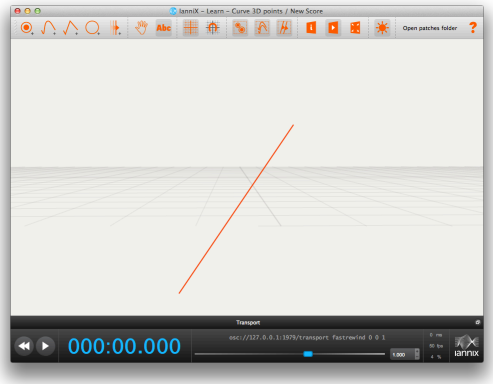


Figure 7. Un segment de droite dans IanniX

2.2.2. *Ellipses*

Une section de courbe en ellipse est définie en fonction du temps par deux rayons r_1 et r_2 et une coordonnée z fixe. Le périmètre de l'ellipse est approximé par :

$$L_k = \pi\sqrt{2({r_1}^2 + {r_2}^2)} \quad (4)$$

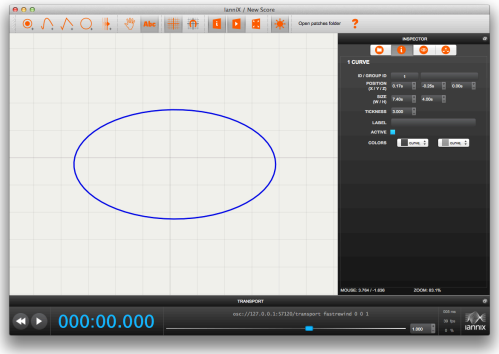


Figure 8. Une ellipse dans IanniX

2.2.3. *Courbes de Bézier*

Une section de courbe en courbe de Bézier cubique est définie en fonction du temps par un point de départ $P_{k,0}$ et un point d'arrivée $P_{k,3}$ ainsi que deux points de contrôle $P_{k,1}$ et $P_{k,2}$:

$$S_k(t) = P_{k,0} \cdot (1-t)^3 + 3 \cdot P_{k,1} \cdot t \cdot (1-t)^2 + 3 \cdot P_{k,2} \cdot t^2 \cdot (1-t) + P_{k,3} \cdot t^3,\ t \in [0,1] \quad (5)$$

La longueur L_k est *approximée* en subdivisant la courbe en segments linéaires.

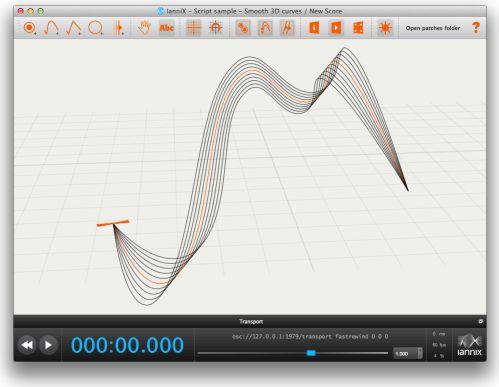


Figure 9. Courbes de Bézier 3D dans IanniX

2.2.4. *Formats vectoriels*

Avec cette palette de 3 sections de courbes différentes (ligne, cercle, Bézier du 3ème ordre), toutes les courbes vectorielles répertoriées dans le format SVG[4] sont possibles.

[4] Scalable Vector Graphics : langage balisé décrivant des graphismes vectoriels bidimensionnels

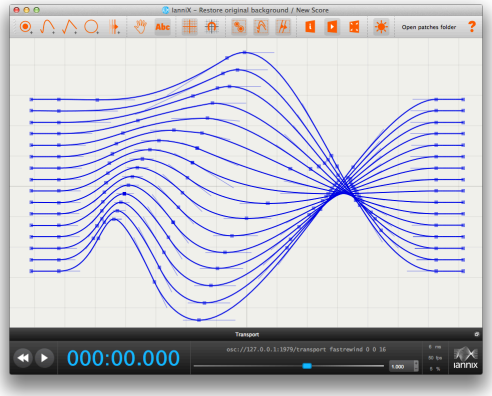


Figure 10. Logo IanniX importé au format vectoriel

2.3. Valeurs : les curseurs

Un ou plusieurs curseurs C peuvent progresser sur une courbe S et exploitent le paramètre t de la courbe. Les coordonnées 3D d'un curseur sont définies par :

$$C = S(Z(t)),\ t \in [0, L_n] \quad (6)$$

Les propriétés des curseurs dans IanniX :

- les curseurs peuvent envoyer des messages indiquant leur position dans l'espace ou leur progression ;
- les curseurs peuvent déclencher des *triggers* sur leur passage ;
- les curseurs peuvent envoyer des messages lorsqu'ils créent un point d'intersection avec une autre courbe.

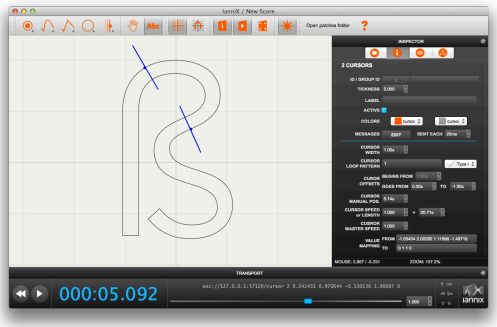


Figure 11. Deux curseurs sur une courbe

IanniX implémente un mécanisme dit de *polytemporalité*, qui permet à chaque curseur de gérer indépendamment sa progression, donc sa relation espace-temps. Celle-ci est réalisée en appliquant une fonction de transfert Z propre à chaque curseur sur le temps, permettant par exemple :

- de progresser normalement sur une courbe si $Z(t) = t$ ou à rebours avec $Z(t) = L_n - t$;
- de doubler la vitesse du curseur sur une courbe, par exemple si $Z(t) = 2 \cdot t$;
- de jouer en boucle une courbe, grâce au reste de la division euclidienne dans les réels ;
- de créer des accélérations et décélérations.

2.4. Temps : le scheduler

IanniX étant un outil informatique, le temps ne peut être vu de manière continue ; il est évidemment échantillonné. La fréquence d'échantillonnage doit être ajustable par l'utilisateur selon ses besoins, mais une valeur comprise entre 200 Hz (*T=5 ms*) et 1 kHz (*T=1 ms*) s'avère suffisante pour couvrir la majorité des usages. L'échantillonnage du temps introduit cependant une difficulté quant au déclenchement des *triggers* :

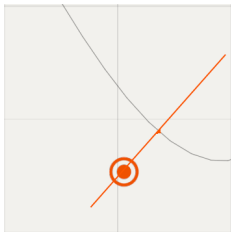


Figure 12. Situation idéale où le curseur "percute" le trigger

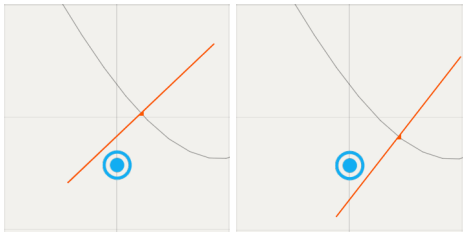


Figure 13. Situation où l'échantillonnage n'est pas suffisant pour déclencher le trigger

IanniX implémente l'interpolation suivante : un curseur déclenche les *triggers* situés entre la position au coup *n* et la position au coup *n-1*.

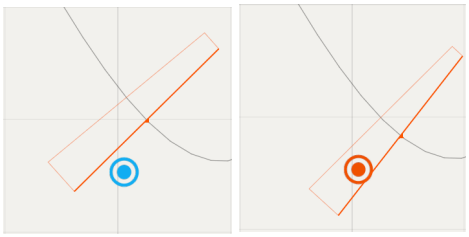


Figure 14. Interpolation des curseurs

3. INTERFACES & MESSAGES

3.1. Interfaçages

3.1.1. Interfaces réseau

La volonté de IanniX est de s'intégrer dans le maximum de contextes applicatifs. Ses interfaces de communications bidirectionnelles sont :

- l'Open Sound Control (OSC) et l'UDP brut, pour les applications temps réel ;
- le MIDI, pour les contextes de contrôle simples ou pour l'interfaçage sur des DAW[5] ;
- le RS232 , pour le contrôle de hardware ;
- l'HTTP, pour le monitoring sur page Web.

[5] Digital Audio Workstation : station dédiée à l'audio numérique

3.1.2. Interface virtuelle ou directe

Une boucle locale a également été développée, afin que IanniX puisse s'envoyer directement des messages, sans consommation de bande passante réseau ni encapsulation dans un protocole.

Nous reviendrons sur cette notion de *récursivité* à la sous-section 5.3 Écriture récursive.

3.1.3. Interface script

Des commandes JavaScript permettent également de générer des messages sous forme de code / script.

Nous détaillerons cette approche dans le chapitre *3.3.* Partitions génératives.

3.1.4. Interface graphique utilisateur

Enfin, la grande majorité des actions sont réalisables depuis une interface graphique utilisateur soignée et organisée. L'interface graphique a été conçue en quatre zones d'interactions :

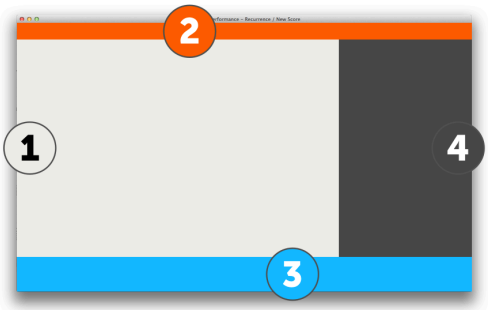


Figure 15. Zoning de l'interface graphique

1. VISUALISATION : représentation graphique de la partition ;
2. CREATION : zone de manipulation de l'espace et outils de création d'objets ;
3. DEROULEMENT : macro-réglages et transport ;
4. INSPECTION : recherche d'objets, propriétés, logs.

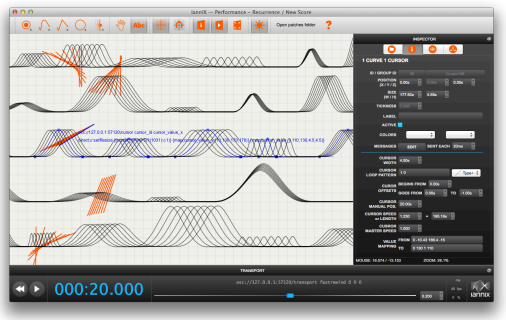


Figure 16. Interface graphique de IanniX

3.2. Gestion par messages

3.2.1. Messages sortants

Les objets de IanniX émettent des messages (les *triggers* en émettent lorsqu'ils sont déclenchés, les *curseurs* en émettent périodiquement lorsqu'ils progressent sur une courbe). Les interfaces et protocoles étant variés (OSC, HTTP, MIDI, RS232[6]…), une homogénéisation du formatage des messages a été imaginée dans IanniX.

Un message IanniX respecte systématiquement un format d'URL :

```
protocole://destination/adresse arguments
```

- **`protocole`** définit l'interface à utiliser (**`osc`**, **`udp`**, **`midi`**, **`serial`**, **`http`**, **`direct`**) ;
- **`destination`** décrit en Open Sound Control, UDP et HTTP, l'IP et le port ; en MIDI ou port série, le nom du port / dispositif de sortie ;
- **`adresse`** représente l'adressage au sein du protocole choisi : en Open Sound Control, l'adresse OSC ; en MIDI le contrôle à moduler : control change (**cc**), note on/off (**note**), program change (**`pgm`**), pitch bend (**`bend`**) ;

Les **`arguments`** sont des variables que IanniX adapte en temps réel en fonction de la partition. Il existe une cinquantaine d'arguments possibles ; quelques exemples[7] :

- **`cursor_zPos`** donne l'élévation *z* d'un curseur ;
- **`trigger_xPos`** retourne la position en *x* d'un *trigger* déclenché ;
- **`collision_yPos`** fournit la coordonnée *y* du point d'intersection entre un curseur et une courbe ;
- **`cursor_angle`** représente l'angle du curseur sur la courbe ;
- **`trigger_distance`** calcule la distance entre le *trigger* déclenché et le point sur la courbe du curseur.

Concrètement, un message IanniX typique serait :

```
osc://192.168.0.10:57120/synth cursor_id
cursor_xPos cursor_yPos
```

Cet exemple envoie sur l'IP **`192.168.0.10`** et le port **`57120`** un message Open Sound Control à l'adresse **`/synth`** avec 3 arguments :

- l'ID du curseur qui envoie le message (*entier*) ;
- la position *x* du curseur (*flottant*) ;
- la position *y* du curseur (*flottant*).

[6] Protocole série couramment utilisé pour dialoguer avec des interfaces matérielles

[7] Documentation complète : http://iannix.org/fr/documentation

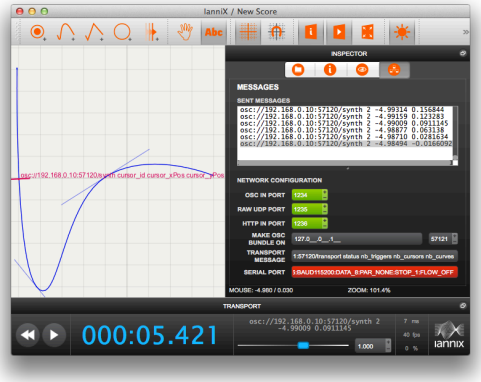


Figure 17. Log des envois dans IanniX

3.2.2. Commandes entrantes

Des commandes entrantes sont possibles, afin de contrôler IanniX depuis une autre application.

Toute la composition de partition repose sur des messages entrants, qu'ils proviennent de l'interface graphique *(les messages ne sont pas visibles par l'utilisateur)*, des interfaces réseau, de l'interface virtuelle ou de scripts.

Par exemple, le message Open Sound Control suivant **`/iannix/add trigger 5`** va ajouter un *trigger* dont l'ID vaut 5 dans la partition ; ou encore le message **`play 0.4`** va jouer la partition avec un facteur de vitesse de 0.4.

Il existe plus d'une soixantaine de commandes possibles, répertoriées dans une API[8] documentée.

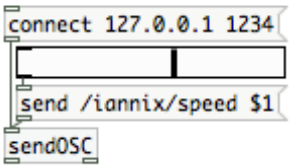


Figure 18. Contrôle de la vitesse de lecture en temps réel depuis PureData

3.3. Partitions génératives

IanniX peut interpréter des scripts JavaScript grâce à son moteur basé sur ECMAScript (norme ECMA-262[9]).

3.3.1. Langage de programmation

Les contraintes de programmation sont assez simples : un script IanniX doit obligatoirement contenir une fonction **`onCreate()`** qui est appelée au moment de la génération de la partition (à l'ouverture du fichier). Le code est ensuite interprété comme un script JavaScript habituel.

[8] Application Programming Interface : interface fournie pour piloter un programme informatique

[9] http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf

Pour envoyer un message à IanniX, une commande spéciale **`run(message)`** a été mise en place. Par exemple, pour ajouter une courbe d'ID = 9, on saisira :

```
run("add curve 9");
```

IanniX apporte cependant quelques enrichissements, notamment pour simplifier le *mapping* de valeurs d'un intervalle à un autre, ou encore quelques fonctions mathématiques usuelles (nombre aléatoire entre deux bornes…).

3.3.2. *Exemples de mise en œuvre*

En exploitant les boucles ou la récursivité, il est possible de générer des partitions graphiques complexes ou mathématiques. Exemple d'un script :

```
function onCreate() {
 //Boucle de 0 à 24
 for(var index=0 ; index < 25 ; index++) {
  //Crée une courbe d'ID 1000, 1001...
  run("add curve " + (1000 + index));
  //Premier point de la courbe
  run("setPointAt current 0 "
                          + (-index) + " 0");
  //Deuxième point de la courbe
  run("setPointAt current 1 0 "
                        + (24-index) + " 0");

  //Crée un curseur d'ID 0, 1, 2...
  run("add cursor " + index);
  //Lie le curseur à la courbe précédente
  run("setCurve current lastCurve");
  //Fixe la durée de parcours à 5 sec
  run("setSpeed current auto 5");
 }
}
```

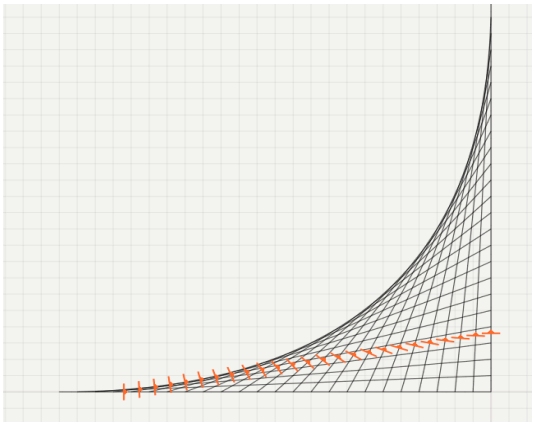


Figure 19. Résultat du script ci-dessus

Quelques autres exemples :

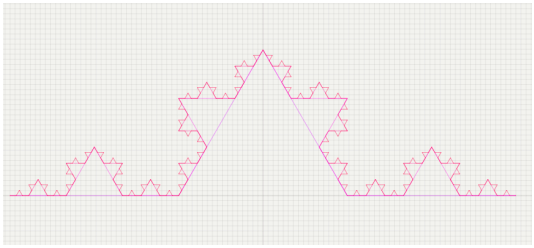


Figure 20. Génération d'une partition à base de fractales

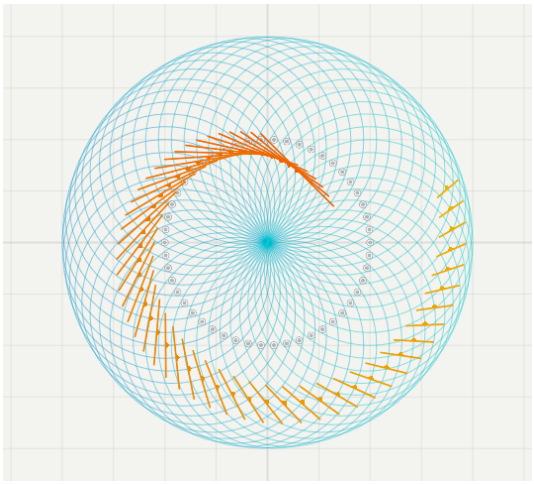


Figure 21. Script générant des cercles

3.3.3. *In-message programming*

Il est aussi possible d'insérer directement du code JavaScript dans les messages émis par les objets IanniX. Pour réaliser cette opération, on insère le code entre accolades.

```
osc://192.168.0.10:57120/synth
        cursor_id
{cursor_xPos + cursor_yPos}
      {random(20,100)}
```

Cet exemple envoie sur l'IP **`192.168.0.10`** et le port **`57120`** un message Open Sound Control à l'adresse **`/synth`** avec 3 arguments :

- l'ID du curseur qui envoie le message ;
- la somme de la coordonnée *x* et *y* du curseur ;
- un nombre aléatoire entre 20 et 100.

4. LE LOGICIEL

4.1. Architecture logicielle

Depuis 2004, IanniX a été développé avec le *framework* de développement Qt. Ce choix s'avère toujours aussi pertinent pour les versions futures pour les raisons suivantes :

- *framework* open source, compatible avec des modèles de licences types GPL[10] ;
- programmation en C++, performant et stable, avec une précision du temps de 100 µs ;
- *cross-platform*, qui peut s'exécuter sur les trois principaux systèmes d'exploitation Linux, Mac OS et Windows ;
- intégration native d'OpenGL, pour le rendu 2D et 3D de manière fluide et interactive ;
- interfaces graphiques riches avec une expérience utilisateur intuitive, élégante et efficiente.

Le logiciel a été divisé en sept grands modules indépendants, fournissant chacun un certain nombre de services.

[10] General Public License : licence pour les logiciels libres

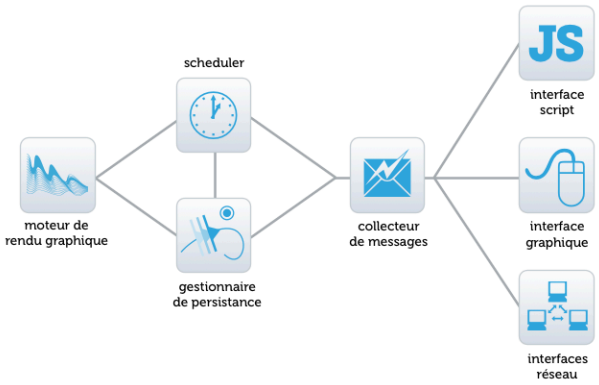


Figure 22. Architecture logicielle

4.2. Performances & optimisations

D'une manière générale, IanniX a été profilé pour deux optimisations :

- le calcul des positions des curseurs sur les courbes, le déclenchement des *triggers* et l'émission de message qui sont réalisés en CPU[11] ;
- l'affichage et l'animation des objets qui est effectuée en GPU[12].

4.2.1. Persistance des données

Pour le stockage des objets et des partitions, chaque référence d'objet est stockée en double :

- une première copie est stockée dans une table de hachage qui permet très rapidement :
  - de parcourir l'ensemble des objets grâce à des itérateurs (pour l'affichage),
  - de trouver un objet par son ID (pour les messages entrants) avec une complexité qui tend vers $O(1)$ [2] ;
- une deuxième copie est stockée dans un arbre qui différentie les types d'objets, leur activité, leur regroupement en groupes et leur position dans l'espace : cette optimisation permet de rechercher très rapidement un ou plusieurs objets en connaissant au préalable ses caractéristiques (pour le calcul).

4.2.2. Représentation graphique

L'intégralité de la représentation graphique des partitions bénéficie de l'accélération graphique OpenGL.

Les courbes sont optimisées pour :

- le calcul de la position d'un curseur sur une courbe est réalisée et optimisés en CPU ;
- le calcul et dessin des courbes se fait intégralement en GPU.

Le *framerate* est fixé par défaut à 40 images par seconde, mais il est ajustable par l'utilisateur selon ses besoins ou ressources disponibles.

4.2.3. Mise en cache de la construction de message

La génération des messages est mise en cache dans IanniX afin d'émettre 1000 messages par seconde avec le minimum de ressources CPU. Au-delà de ce seuil, les optimisations sont négligeables par rapport au coût d'envoi d'un datagramme réseau par le système d'exploitation.

4.3. Modèle de diffusion & de contribution

IanniX est diffusé sur son site Internet (www.iannix.org) en licence GPL 3. Les binaires des versions Linux, Mac OS et Windows sont proposés dans leurs doubles versions 32 bits et 64 bits. Un *repository* Git (www.github.com/iannix/IanniX) permet également aux potentiels contributeurs de participer facilement au développement du logiciel ; la chaîne de compilation ne nécessitant que Qt SDK. Enfin, les utilisateurs peuvent échanger des patchs, astuces, bugs, solutions… dans un forum de discussion.

Outre une refonte logicielle intégrale mi-2011, IanniX a renforcé sa présence sur le web (unique lieu de diffusion) grâce à du *teasing vidéo* et à la création de supports de communication actuels *(Facebook, mailing-list, Twitter)*.

Sur ces neuf derniers mois, IanniX compte 20000 visiteurs uniques sur son site web (avec un taux de rebond très faible de 1.7%), 5200 utilisateurs et une quarantaine de lancements quotidiens de l'application. Deux tiers des utilisateurs sont sur Mac, 30% sur Windows et le reste sur Linux. Seuls 15% des utilisateurs sont français.

La communauté d'utilisateurs a également développé des exemples et tutoriaux pour intégrer IanniX dans divers logiciels (Max4Live, SuperCollider, CSound…).

Enfin, l'association « IanniX » encourage la promotion du logiciel dans tous ses aspects : recherche, création, enseignement, formation, publications, études, développements…

5. NOUVELLES POSSIBILITES D'ECRITURE

5.1. Approche macroscopique et microscopique du temps

Contrairement à l'UPIC qui considérait une seule ligne temporelle macroscopique, IanniX intègre une gestion du temps à l'échelle microscopique (une infinité de lignes de temps possibles). Cette double approche permet d'envisager des compositions à plusieurs échelles de lecture et d'interprétation.

[11] Central Processing Unit : processeur logique d'une machine
[12] Graphics Processing Unit : processeur graphique d'une machine

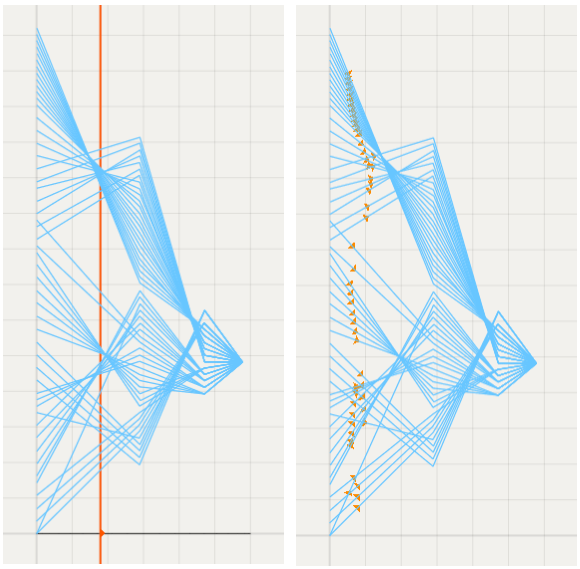


Figure 23. Partition numérisée de Metastasis de Iannis Xenakis jouée sous l'approche macroscopique *(à gauche)* et microscopique *(à droite)*

5.2. Creative & live coding

IanniX est certes un logiciel autonome — et non une librairie — mais son langage de script intégré lui permet de s'inscrire dans la liste des outils de *creative coding* ; l'aspect temps réel lui confère aussi l'appellation de *live coding*. L'intérêt majeur de l'écriture par scripts est d'intégrer l'aléatoire, les boucles, les équations mathématiques et la logique.

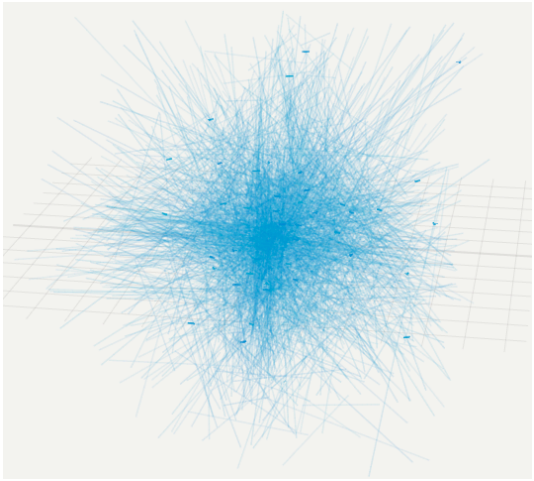


Figure 24. Script générant des lignes aléatoirement dans l'espace, sous contraintes

5.3. Écriture récursive

La fonction principale de IanniX est de générer des messages qui sont envoyés à un environnement dédié. Mais qu'advient-il si IanniX s'envoie des messages à lui-même modifiant ainsi la structure géométrique ou temporelle des partitions ?

Cette nouvelle forme d'écriture, appelée *écriture récursive*, est radicalement innovante et sa complexité la rend encore difficile à maitriser : des phénomènes de stabilisation, d'*exponentialisation*, d'anéantissements ou de chaos se produisent.

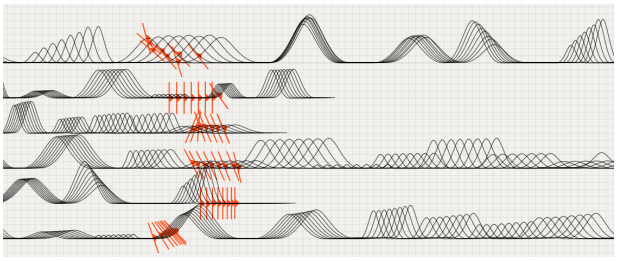


Figure 25. Partition de la performance « Récurrences » de Thierry Coduys au centre DATABAZ d'Angoulême *(partition distribuée librement dans l'application IanniX)*

5.4. Spatialisation 3D

Couplé à un moteur audio de spatialisation (Ambisonic 3D…), la palette d'outils de IanniX permet l'écriture de trajectoires en 3D :

- le *trigger* déclenche un son ;
- la courbe définit la trajectoire du son dans l'espace ;
- le curseur joue sur le temps (accélérations, répétitions, déplacements, etc.)

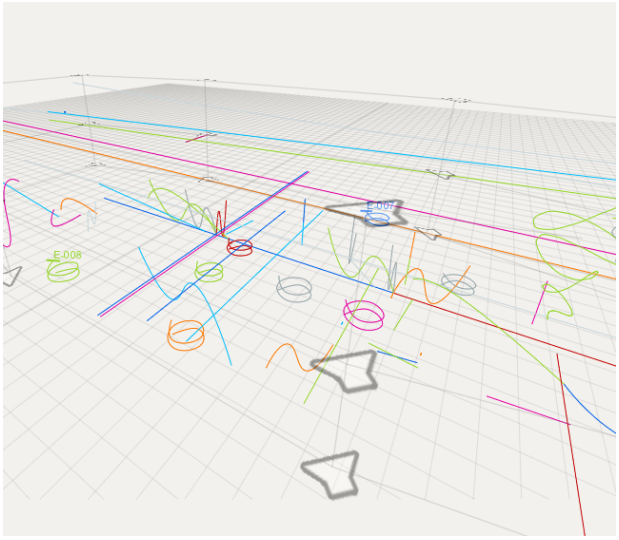


Figure 26. Trajectoires de spatialisation pour l'exposition universelle de Yeosu 2012

5.5. Interfaces temps réel

La versatilité des entrées / sorties de IanniX permettent d'utiliser des capteurs ou actionneurs récents de toutes natures.

L'intérêt d'un tel dispositif de captation réside dans la manipulation, modification ou la génération de courbes en temps réel tel un geste instrumental interprétant une partition.

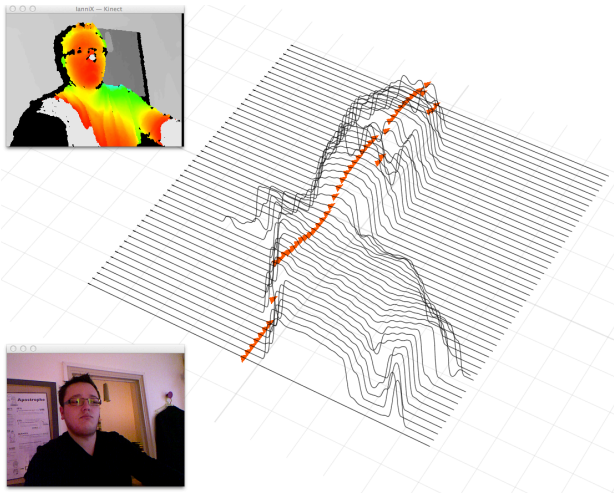


Figure 27. Extrusion des courbes à l'aide d'un capteur Kinect à la manière d'un scanner 3D

6. CONCLUSION & PERSPECTIVES

IanniX offre une approche graphique et 3D du séquencement temps réel à l'aide de trois objets fondamentaux (les triggers, les courbes et les curseurs). Les interfaces implémentées, quelles soient physiques (OSC, MIDI, RS232…), graphiques (interface utilisateur), logicielles (JavaScript) ou virtuelles (récursivité), contrôlent, modifient ou génèrent des partitions de séquencement grâce notamment à un système d'abstraction par messages uniques.

Les travaux menés pour IanniX 0.8 depuis mai 2011 offrent à l'outil des performances accrues et une meilleure stabilité. Une communauté d'utilisateurs s'est rapidement créée et un modèle open source contributif a été mis en route.

De nouvelles formes d'écritures sont apparues au fil du développement et des projets réalisés[13] avec IanniX, notamment l'écriture récursive, l'intégration du geste instrumental et le *live coding*.

Les prochains développements de IanniX porteront notamment sur :

- les courbures de l'espace-temps, afin de conférer à IanniX des propriétés physiques relativistes ;
- l'intégration de propriétés physiques aux objets (masses, ressorts, modèles physiques…) ;
- un passage progressif d'une logique booléenne vers une logique floue ;
- la possibilité de visualiser une partition sous différentes représentations symboliques (feuilles de style).

Au-delà des aspects logiciels, la dissémination des travaux autour de IanniX est un objectif fort, aussi bien dans le cadre scientifique (publications), universitaires (workshop et utilisation dans des formations) ou intellectuel (accompagner les créations).

[13] Exemples de projets : http://iannix.org/fr/projects.php

7. REFERENCES

[1] TimelinerSA, http://vvvv.org/contribution/timelinersa

[2] Algorithmic Complexity of Container Classes, *Qt Documentation*, http ://qt-project.org/doc/qt-4.8/containers.html

[3] Henry, D. "PTL, a new sequencer dedicated to graphical scores", International Computer Music Conference Proceedings, Miami, USA, 2004

[4] Virage, http://www.virage-platform.org/

[5] AlgoScore, http://kymatica.com/Software/AlgoScore

[6] Wright, M. et Freed, A. "Open Sound Control: A New Protocol for Communicating with Sound Synthesizers" Proceedings of the International Computer Music Conference 1997, Thessaloniki, Hellas, pp. 101-104.

[7] Coduys, T. et Ferry G., "IanniX, aesthetical / symbolic visualisations for hypermedia composition", Sound and Music Computing, 2004

LE COMPILATEUR EN LIGNE DE FAUST : UN IDE EN LIGNE POUR LE LANGAGE DE PROGRAMMATION FAUST

Romain MICHON et Yann ORLAREY
GRAME, Centre national de création musicale
9 rue du Garet
69202 Lyon,
France,
rmnmichon@gmail.com et orlarey@grame.fr

RÉSUMÉ

FAUST est un langage de programmation fonctionnel pour le traitement du signal et la synthèse de sons en temps réel. Grâce à un système de fichiers d'architectures, un seul et unique programme FAUST peut être utilisé pour générer du code pour un ensemble de types d'applications et de plug-ins.

Le compilateur en ligne de FAUST ici présenté est une application Web écrite en `PHP` et en `JavaScript` offrant un environnement de développement multiplateforme et multiprocesseur pour le langage FAUST. Cet outil rend possible l'utilisation de la plupart des fonctionnalités de FAUST dans un navigateur Web et intègre un catalogue d'exemples évolutif faisant de lui une plate-forme pour utiliser et échanger facilement tout objet FAUST.

Le fonctionnement du compilateur en ligne de FAUST est présenté en détail dans cet article. Les possibilités offertes par cet outil sont discutées et une brève ouverture sur les enjeux de l'utilisation des technologies Web pour l'informatique musicale est faite.

1. INTRODUCTION

FAUST [4] est un langage de programmation fonctionnel pour le traitement du signal et la synthèse de sons en temps réel. Grâce à un système de fichiers d'architectures, un seul et unique programme FAUST peut être utilisé pour générer du code pour un ensemble de types d'applications et de plug-ins.

Le compilateur en ligne de FAUST est une application Web écrite en `PHP` et en `JavaScript` offrant un environnement de développement multi-plateforme et multiprocesseur pour le langage FAUST. Cet outil rend possible l'utilisation de la plupart des fonctionnalités de FAUST dans un navigateur Web. Nous pensons qu'il permet de résoudre l'une des principales limites de FAUST : le nombre important de dépendances requises pour l'utilisation de ses architectures. En effet, bien que FAUST soit totalement indépendant de toute bibliothèque externe pour pouvoir être compilé, l'utilisation de certaines de ses architectures se montre assez complexes à cause du grand nombre de dépendances sur lesquelles leur fonctionnement repose. Le problème des architectures système que certains utilisateurs travaillant par exemple sous Windows ont pu rencontrer est également solutionné.

Le compilateur en ligne de FAUST intègre un catalogue d'exemples éditable par ses utilisateurs ce qui en fait également une plate-forme d'échange. Grâce à l'interaction entre ses différents éléments, il peut être vu aussi comme un outil pédagogique dans lequel il est possible de visualiser un algorithme de traitement du signal écrit en FAUST, son équivalent en `C++`, sa représentation sous forme de digramme, sa documentation et son résultat.

Le compilateur en ligne de FAUST a été implémenté dans le cadre du développement du nouveau site internet de FAUST [1] . Il peut être intégré dans toute page `HTML` et installé facilement sur un serveur Apache.

2. INTERFACE

Le compilateur en ligne de FAUST est basé sur un block de taille fixe dont les proportions ont été définies en fonction de la taille du nouveau site Web de FAUST. Toutefois, il peut être aisément mis à l'échelle et intégré dans n'importe quelle `div` d'une page `HTML` en utilisant un système de `iframe` (Cf. 4). Un mode "plein écran" est également disponible afin d'offrir aux utilisateurs une zone de travail plus large.

La navigation entre les différents états d'un objet FAUST se fait au travers d'un ensemble d'onglets accessibles à l'aide d'une barre horizontale (Cf. cadre n° 1 de la figure 1).

2.1. Editer le code

Le code FAUST peut être édité de deux manières différentes dans le compilateur en ligne. La plus évidente consiste à utiliser l'éditeur de code placé dans l'onglet `Faust Code`. Le programme `JavaScript` CodeMirror [2] est utilisé pour mener à bien cette tâche. Ce dernier a été choisi pour sa compatibilité avec un grand nombre de navigateurs Web (Firefox, Safari, Opera, Internet Ex-

1. `http://faust.grame.fr`
2. `http://codemirror.net`

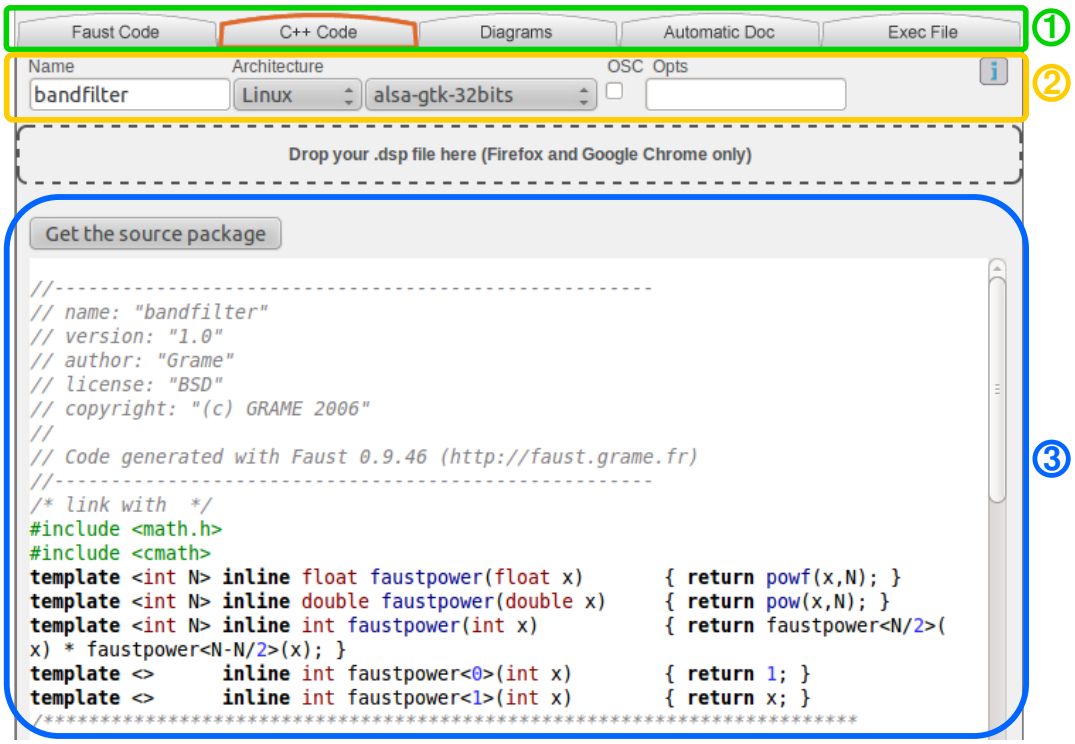


Figure 1. Vue d'ensemble des différentes zones constitutives du compilateur en ligne de FAUST. Le cadre 1 contient la barre de navigation, le cadre 2 le menu des options de compilation et le cadre 3 les différents types de résultats demandés ou l'éditeur de codes affichés en fonction de l'onglet sélectionné.

plorer, Google Chrome, etc.). Il utilise un modèle de coloration de code spécifique à FAUST.

Grâce a un script `AJAX`, le compilateur en ligne de FAUST autorise également le glissé/déposé de tout fichier `.dsp` contenant un code FAUST dans une zone placée juste au dessus de l'éditeur de code (Cf. figure 1). Cette zone est présente dans chaque onglet. Cela permet donc de compiler un code FAUST créé en local rapidement et de manière très interactive.

2.2. Compilation

2.2.1. *Options de compilation*

Le menu d'options de compilation est visible dans les onglets `C++ Code` et `Exec File`. Il est situé juste en dessous de la barre de navigation (cf. cadre 2 sur la figure 1) et permet de définir le type d'exécutable qui sera généré lors de la compilation `C++`. La plupart des architectures FAUST sont disponibles et une option "OSC" peut être sélectionnée si l'architecture choisie le permet. D'autres options peuvent être passées en argument à l'aide de la zone de texte `Opts`.

Le serveur Linux (Ubuntu) sur lequel est installé le compilateur en ligne de FAUST à GRAME permet de compiler le code `C++` généré par FAUST en 32 bits et en 64 bits pour les plate-formes suivantes :

– Linux ;
– Windows, en utilisant le cross compilateur `MinGW` [3] ;
– OSX 10.6, en effectuant la compilation sur un serveur Mac distant à l'aide du système d'exécution de commandes de `SSH` [4] et en important le fichier obtenu avec `scp`.

Si l'une de ces options est changée dans les onglets `C++ Code` ou `Exec File`, le résultat est automatiquement mis à jour.

2.2.2. *Compilation*

Afin de mener à bien les différentes étapes de compilation, un dossier temporaire contenant le fichier FAUST a traiter et un Makefile généré en fonction des options et de l'architecture choisies par l'utilisateur est créé sur le serveur. Il sera utilisé pour produire le code `C++`, les diagrammes, la documentation de l'objet, l'exécutable et les différents packages.

2.3. Générer le code C++

Lorsque l'onglet `C++ Code` est sélectionné, le compilateur de FAUST est appelé sur le serveur par le Makefile. Le code généré est alors traité par `Highlight` [5] pour le colorer. Si une architecture autre que *none* est sélectionnée par un utilisateur, le Makefile créera également

3. `http://www.mingw.org`
4. `http://www.openssh.com`
5. `http://www.andre-simon.de/doku/highlight/en/highlight.html`

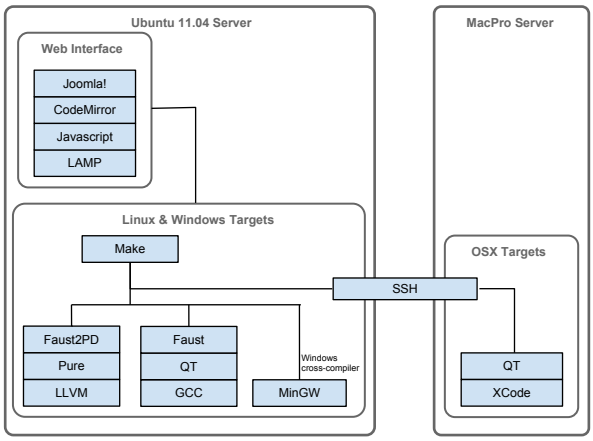


Figure 2. Schéma des différents composants du serveur.

un package contenant tous les éléments nécessaires pour qu'il puisse compiler le code `C++` sur sa machine sans que FAUST n'y soit installé. Il est toutefois important de noter qui si l'option `OSC` est sélectionnée, le package généré lors de l'étape décrite précédemment ne sera compilable que si FAUST est installé sur le système de l'utilisateur car les bibliothèques alors requises font partie de la distribution de FAUST.

Si le code FAUST entré contient des erreurs, le message retourné par le compilateur sera affiché au lieu du code `C++`. Le compilateur en ligne réagira de la même manière dans le cas des onglets `Diagrams` et `Automatic Doc`.

2.4. Visualiser les différents types de diagrammes

Plusieurs types de diagrammes du code FAUST donné au compilateur en ligne peuvent être visualisés dans l'onglet `Diagrams` :

- block-diagramme (Cf. figure 3) ;
- schéma des signaux (Cf. figure 4) ;
- schéma des tâches.

Comme le montrent les figures 3 et 4, le type de diagramme affiché est sélectionné à l'aide d'un menu déroulant. Les images générées sont dans un format vectoriel (SVG), il est donc possible de zoomer sur ces dernières sans voir apparaître de crénelage. De plus, les block-diagrammes peuvent être parcourus en cliquant sur leurs différentes boîtes.

2.5. Visualiser la documentation générée automatiquement d'un objet FAUST

Grâce a de récents travaux menés à GRAME dans le cadre du projet ASTREE (voir *Remerciements*), le compilateur de FAUST est maintenant en mesure de générer automatiquement la documentation mathématique d'un objet écrit avec le langage FAUST [1]. Le compilateur en ligne est compatible avec cette fonction et permet d'afficher le fichier `pdf` produit à l'aide de Google Document

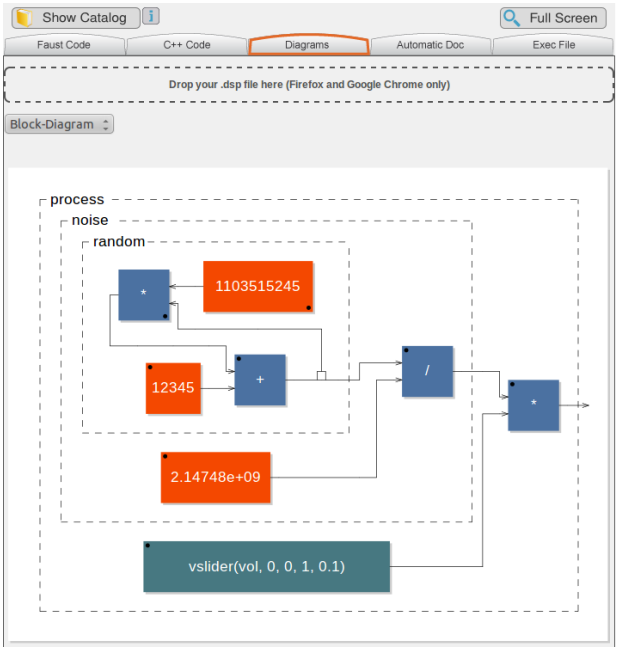


Figure 3. Capture d'écran de l'onglet `Diagram`. L'option *Block-Diagram* est sélectionnée.

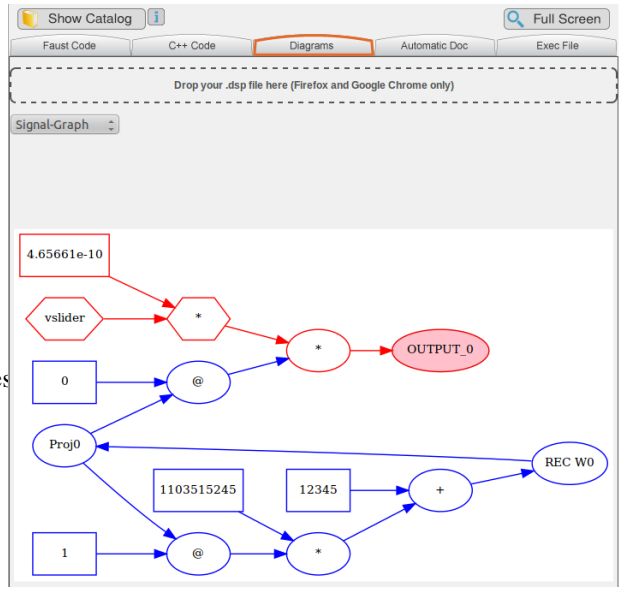


Figure 4. Capture d'écran de l'onglet `Diagram`. L'option *Signal-Graph* est sélectionnée.

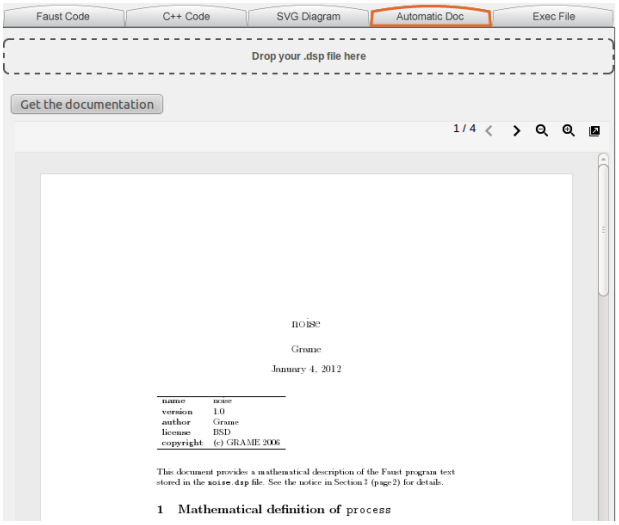


Figure 5. Capture d'écran de l'onglet `Automatic Doc`.

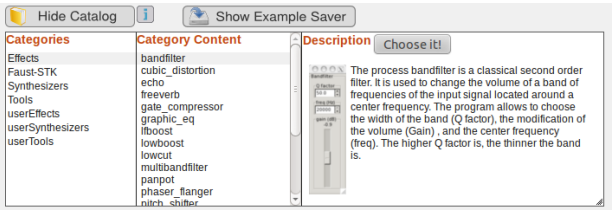


Figure 7. Le catalogue d'exemples.

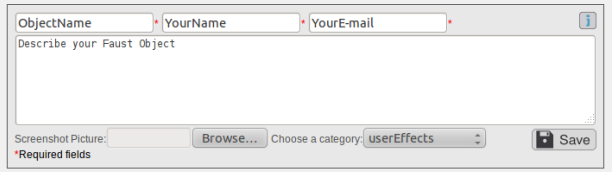


Figure 8. L'enregistreur d'exemples.

Viewer [6]. Le Makefile décrit dans 2.3 est utilisé pour mener à bien cette opération.

Une capture d'écran de l'onglet `Automatic Doc` est visible dans la figure 5.

2.6. Générer un exécutable

Lorsque l'onglet `Exec File` est choisi, la compilation `C++` est effectuée en fonction du système d'exploitation et de l'architecture processeur sélectionnée. Si cette opération s'est déroulée correctement, un bouton de téléchargement apparaît. Sinon, le message d'erreur retourné par le compilateur `C++` est affiché. Le fichier généré sera, en fonction de l'architecture FAUST choisie, soit un paquet contenant plusieurs fichiers, soit un exécutable.

3. CATALOGUE D'EXEMPLES

Le compilateur en ligne de FAUST contient une plate-forme pour échanger facilement et de manière interactive des objets FAUST : le catalogue d'exemples. Deux éléments le constituent : le catalogue lui-même et un enregistreur d'exemples.

3.1. Le catalogue

La catalogue à la forme d'un explorateur de dossiers dans lequel chaque code FAUST est placé dans différentes catégories (Cf. figure 7) :

- Effects ;
- Faust-STK [3] ;
- Synthesizers ;
- Tools.

Un ensemble de catégories modifiables "utilisateur" (user) sont également disponibles.

6. `http://docs.google.com/viewer`

Lorsqu'un objet FAUST est sélectionné, une courte description et une capture d'écran de ce dernier sont affichées dans la colonne `Description`. Il peut alors être envoyé au compilateur en ligne soit en le double-cliquant, soit poussant le bouton `Choose it!`.

La catalogue d'exemples peut être affiché dans tous les onglets du compilateur en ligne de FAUST. Ainsi, un utilisateur peut choisir de visualiser simplement le diagramme d'un objet ou sa documentation.

3.2. Enregistrer un objets Faust dans le catalogue

Comme cela a été dit précédemment, le compilateur en ligne de FAUST se veut être une plate-forme pour l'échange d'objet FAUST. Ceci est rendu possible par l'enregistreur d'exemples qui permet à un utilisateur de sauvegarder son code FAUST dans une des catégories "utilisateur" du catalogue. Aucune identification n'est requise pour effectuer cette tâche : n'importe qui peut enregistrer son travail sur le catalogue à condition qu'il s'agisse d'un code FAUST compilable correct. Lorsqu'un nouvel exemple est placé dans le catalogue, l'utilisateur peut choisir d'y ajouter une courte description ainsi qu'une capture d'écran.

Si un exemple existant est modifié par un utilisateur différent de celui qui l'a créé, chaque utilisateur qui a contribué à cet exemple sera informé de cette modification par un e-mail. Enfin, n'importe qui peut choisir de supprimer un exemple contenu dans le catalogue. Tout comme dans le cas précédent, l'ensemble des contributeurs de cet objet sont alors avertis.

Dans le but de conserver un bon ordre dans le catalogue et d'éviter les spams et les duplicatas, un système de vote a été implémenté. Cet outil peut aussi permettre de mettre en valeur les meilleurs exemples utilisateurs.

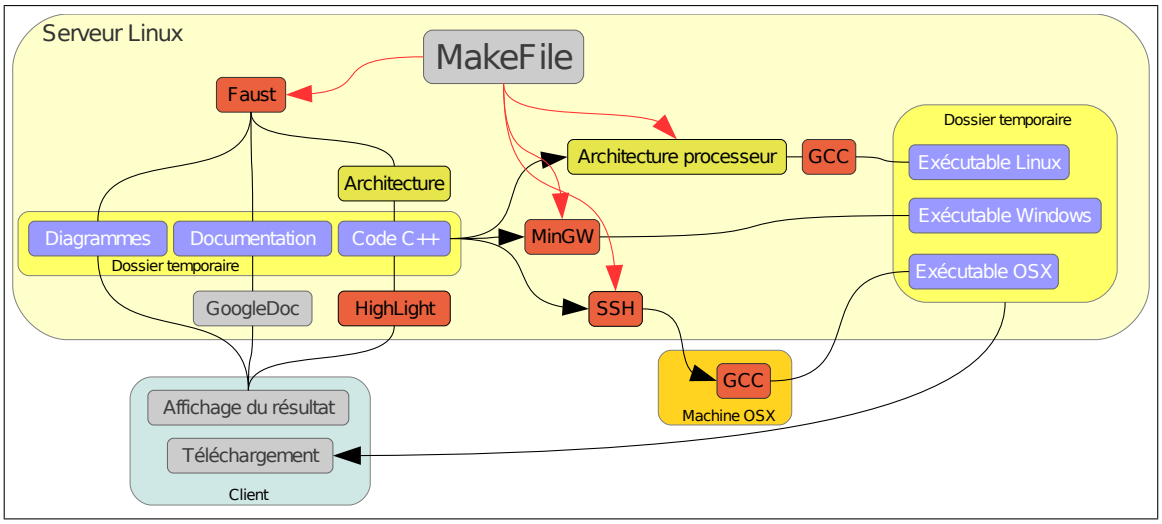


Figure 6. Fonctionnement du compilateur en ligne de FAUST.

4. EXPORTER LE COMPILATEUR EN LIGNE DE FAUST

Le compilateur en ligne de FAUST peut être implanté très facilement dans un page HTML en utilisant une `iframe` :

```
<iframe src="http://faust.grame.fr/
    compiler" style="border:none" height="
    1200px" width="722px"></iframe>
```

Il peut aussi est installé sur un serveur Apache [7] simplement en copiant son code source, disponible dans la distribution de FAUST [8].

5. TRAVAUX FUTURS

De récentes améliorations apportées au compilateur de FAUST lui permettent de générer du code JAVA et LLVM [9] en plus de C++. Ainsi, il pourrait être intéressant d'implémenter une architecture FAUST qui permette de générer des applets JAVA directement utilisables en ligne depuis un navigateur Web. Nous espérons donc pouvoir compléter assez rapidement le compilateur en ligne de FAUST avec un nouvel onglet dans lequel un applet JAVA correspondant au code FAUST édité serait utilisable. Nous avons bon espoir d'implémenter un système dans lequel les changements apportés au code seraient visibles presque immédiatement au niveau de l'applet JAVA qui resterait active jusqu'à la fermeture de la session. Par conséquent, une forme de "live coding" pour FAUST deviendrait disponible.

Le compilateur en ligne de FAUST semble offrir une bonne solution aux problèmes de plate-formes et de dépendances impliqués par l'utilisation de certains fichiers d'architecture. Il pourrait ainsi être intéressant de mettre en place un système de service Web RESTful[5] pour FAUST accessible à l'aide d'un API simple. Le programme FaustWorks [10] pourrait alors utilisé un tel système pour offrir la possibilité aux utilisateurs de compiler leur code FAUST pour des plate-formes différentes de la leur et avec des architectures qu'ils n'arrivaient pas a utiliser sur leur système.

7. http://www.apache.org
8. http://sourceforge.net/projects/faudiostream
9. http://llvm.org/

6. CONCLUSION

Le compilateur en ligne de FAUST simplifie significativement l'utilisation de FAUST en mettant à disposition un environnement de programmation prêt à l'emploi et libre de tout problèmes de compatibilité et d'installation.

Il est certain que les technologies Web jouent un rôle de plus en plus important dans le domaine de l'informatique musicale. L'avènement de nouveaux standards tel que HTML5 ou la nouvelle API web-audio [11] ouvre la voie vers de nouveaux domaines d'exploration tel que le traitement numérique du signal en ligne.

7. REMERCIEMENTS

Ce travail a été effectué dans le cadre du projet ASTREE (ANR-08-CORD-003) sur la préservation des pièces de musique électronique en temps réel.

8. REFERENCES

[1] Barkati, K., Fober D., Letz, S. et Orlarey, Y. "Two recent extensions to the Faust compiler", *Proceedings of the Linux Audio Conference (LAC-2011)*, National University of Ireland, Maynooth, Irlande, 2011.

10. FaustWorks est un IDE pour FAUST implémenté avec Qt.
11. https://dvcs.w3.org/hg/audio/raw-file/tip/webaudio/specification.html

[2] Fober, D., Orlarey, Y. et Letz, S. "Faust architecture design and OSC support", *Proceedings of the Conference on Digital Audio Effects (DAFx-11)*, IRCAM, Paris, France, 2011.

[3] Michon, R. et Smith, J. "Faust-STK : a set of linear and nonlinear physical models for the Faust programming language", *Proceedings of the Conference on Digital Audio Effects (DAFx-11)*, IRCAM, Paris, France, 2011.

[4] Orlarey, Y., Fober, D. et Letz, S. "An algebra for block diagram languages", *Proceedings of the International Computer Music Conference (ICMA)*, Gothenburg, Suède, 2002.

[5] Richardson, L. et Rubu, Y. *Services Web RESTful*. O'Reilly, Paris, 2007.

UNE INTERFACE GRAPHIQUE DE MANIPULATION D'UNITES MODULAIRES DANS SUPERCOLLIDER

Frédéric Dufeu
Université Rennes 2
frederic.dufeu@gmail.com

RÉSUMÉ

La modularité caractérise une part importante des lutheries développées à partir des nouvelles technologies. Parmi d'autres langages courants d'informatique musicale, SuperCollider de James McCartney est largement adapté à la création d'unités de génération et de traitement de signal, ainsi qu'à leur assemblage selon une approche modulaire dynamique. Pourtant, lorsque des unités élémentaires sont prédéfinies tandis que leurs interconnexions sont laissées ouvertes, l'établissement d'une configuration particulière implique un important travail de codage, qui rend l'expérimentation peu flexible. Cet article présente un programme permettant, d'une part, de décrire sous forme de code des modules autonomes et, d'autre part, de les interconnecter de manière efficace et pratiquement immédiate à partir d'une interface graphique programmée en OpenGL dans Max/MSP/Jitter. À partir de types d'unités définis par l'utilisateur, cette interface permet de créer un nombre indéfini d'instances et de les interconnecter selon n'importe quelle configuration, sans aucune limitation liée à leurs caractéristiques techniques. Moyennant une préparation des modules et leur intégration au programme général, ce programme favorise l'exploration intuitive des ressources musicales et sonores de SuperCollider.

1. INTRODUCTION

La notion de modularité accompagne dans une large mesure l'émergence et les développements d'outils technologiques destinés à la création et à l'expression musicale. Dans un article présentant les nouvelles lutheries selon une approche historique, Jean-Claude Risset parle de « boîte à outils logicielle » pour désigner les programmes Music n de Max Mathews. À partir de Music III (1959), l'utilisateur

> « est laissé libre de décider du type de synthèse sonore qu'il veut mettre en œuvre : il choisit des modules dont chacun correspond à une production ou une transformation sonore […], et il les assemble à loisir, comme s'il "patchait" un synthétiseur modulaire […]. »[1]

Si la libre interconnexion de générateurs ou traitements autonomes permet au musicien de composer son instrument[2], différents niveaux d'ouverture doivent être distingués selon l'environnement considéré. En premier lieu, un synthétiseur modulaire matériel n'offre qu'un nombre limité de types d'unités (oscillateur, filtre, générateur d'enveloppe, amplificateur) et chacun de ces types est lui-même limité en nombre d'instances[3]. Leur extension par des unités supplémentaires existantes ou par la fabrication de nouveaux types de modules est possible, mais lourde et coûteuse. Avec les logiciels classiques de programmation musicale[4], l'implémentation de générateurs ou traitements inédits n'a de limite que la capacité du développeur à exprimer sa conception musicale ou sonore à travers le langage ou l'environnement choisi et pour un type de module, la quantité d'unités fonctionnant simultanément est contrainte par les performances de l'ordinateur hôte, mais leur création est aisément opérable. Ensuite, et au-delà de la flexibilité relative à la création de modules, les interconnexions de ces derniers ne sont pas toujours entièrement ouvertes : un système donné peut ne se prêter qu'à une partie de l'ensemble des connexions

[1] [15], p. 23. Risset précise immédiatement : « Contrairement à ce que pensent beaucoup, la conception de Mathews ne copie pas celle des synthétiseurs : elle a au contraire inspiré les dispositifs de Moog et Buchla, réalisés analogiquement en tirant parti de la commande par tension… Mais seulement à partir de 1964, alors que Music III a été écrit en 1959. En fait, la conception modulaire de Mathews a marqué la plupart des programmes de synthèse – comme Music 360, Music 11, CMusic, Csound – et des synthétiseurs analogiques ou numériques – comme ARP, DX7, 4A, 4B, 4C, 4X, Syter – aussi bien que des langages de simulation de circuits électroniques, et plus tard un langage de créations d'interactions temps réel comme Max. » *Ibidem*.

[2] Dans un article datant de 1999, Marcelo Wanderley et Philippe Depalle avançaient la notion d'« instrument virtuel ou composé » et écrivaient : « En désolidarisant cause et effet, l'incursion de l'électricité dans la lutherie a profondément modifié la nature du jeu des instruments de musique. Et bien qu'une prodigieuse diversification des moyens de production sonore en ait résulté, ce changement de nature n'a pas encore engendré de stratégies de contrôle permettant de développer la même subtilité et la même ampleur de jeu que celle que l'on trouve pour les instruments traditionnels. » [19], p. 145. Bruno Bossis emploie également l'expression « instrument composé » dans son article « Écriture instrumentale, écriture de l'instrument » : « contrairement à l'instrument traditionnel, l'"instrument composé" n'est pas entièrement défini avant le travail d'écriture par le musicien. Le compositeur prend ainsi en charge une partie du métier de luthier. » [4], p. 120.

[3] Par exemple, l'ARP2600 comprend trois *VCOs* (*Voltage-controlled oscillators* : oscillateurs commandés en tension).

[4] Comme Csound, Max/MSP, PureData ou SuperCollider.

théoriquement possibles. Enfin, l'établissement d'une situation modulaire n'est pas toujours propice à un certain idéal de temps réel. Devoir débrancher et rebrancher un cordon ou une broche empêche le passage instantané d'une configuration à une autre. Par ailleurs, certains logiciels ne permettent pas de modifier le trajet du signal sans interrompre le calcul audionumérique[5].

Livré sous une première version par James McCartney en mars 1996[6], SuperCollider se définit comme « un environnement et un langage de programmation pour la synthèse audio en temps réel et la composition algorithmique »[7]. En plus de la qualité de son rendu sonore, les caractéristiques de ce logiciel libre[8] en font un environnement particulièrement favorable à une approche modulaire dynamique pour la synthèse, la lecture, l'enregistrement et le traitement de signal : comme le décrit son premier auteur,

> « alors même que la synthèse fonctionne, des modules peuvent être créés, détruits et interconnectés différemment, des mémoires d'échantillons être allouées et réaffectées. Des traitements peuvent être créés et connectés dynamiquement au sein d'un trajet de signal existant […]. »[9]

Pour autant, SuperCollider est un langage d'informatique musicale général et la mise en œuvre d'un dispositif inspiré par la modularité des nouvelles lutheries n'en constitue qu'une approche particulière. Si un certain nombre de classes intégrées à la distribution publique et de bibliothèques tierces facilitent la création en cours de jeu de structures musicales ou d'entités audionumériques selon une approche de *live coding*[10], il n'existe à notre connaissance aucun développement permettant à la fois de définir dans SuperCollider une grande variété de types de modules et de les intégrer pleinement dans un environnement ergonomique de création et de manipulation d'interconnexions[11].

L'objectif du projet que nous présentons dans cet article est de favoriser l'exploration des *unit generators*[12] de SuperCollider en permettant à l'utilisateur de les interconnecter librement, et sans explicitement coder les opérations habituellement associées à cette tâche, notamment en termes d'allocation des bus véhiculant les signaux. Étant donné un ensemble de types de modules audionumériques prédéfinis mais extensibles, il s'agit de pouvoir composer aussi simplement que possible des graphes reliant les unités concernées entre elles. L'enjeu est de disposer d'un outil flexible pour l'expérimentation et la composition audionumérique, aussi bien approprié au travail de studio qu'à la performance en temps réel. Actuellement en cours de développement par l'auteur, ce programme est déjà fonctionnel sous forme de prototype et appelle, au-delà de ses améliorations locales, plusieurs perspectives que nous évoquerons en conclusion. Auparavant, notre présentation s'articulera en trois parties : dans un premier temps, seront exposées les fonctions de SuperCollider permettant d'établir sous forme de code des entités audionumériques et leurs interconnexions. Nous décrirons ensuite une interface graphique utilisateur simple implémentée dans Max, par laquelle doivent être opérées les manipulations de ces entités et du graphe les organisant entre elles. Enfin, seront présentés les éléments du programme permettant de lier adéquatement l'interface graphique au serveur de synthèse de SuperCollider.

2. IMPLEMENTATION D'UNITES AUDIONUMERIQUES INTERCONNECTEES DANS SUPERCOLLIDER

2.1. Création d'unités avec les classes SynthDef et Synth

Dans sa version 3[13], SuperCollider repose sur une architecture client-serveur. Le serveur, scsynth, écrit en C++, est le programme dédié à la synthèse et au traitement audionumérique[14]. Il reçoit ses commandes par réseau depuis un ou plusieurs clients, le plus couramment celui constituant l'autre versant de SuperCollider : l'application sclang, l'interpréteur dans lequel le code est écrit et exécuté.

> « Typiquement, l'interpréteur traduit le code, écrit en langage SuperCollider, en messages OSC[15] destinés au serveur. L'utilisateur écrit en quelque sorte de la poésie en SuperCollider, qui est ensuite

[5] Jusqu'à récemment, il n'était pas possible dans Max de créer ou détruire un objet ou un cordon MSP sans interruption du calcul audio ; les modifications de configuration modulaire nécessitaient la mise en œuvre de systèmes matriciels. Cette limitation a été supprimée dans la version 6 de Max, commercialisée en octobre 2011 [11].

[6] Pour une description de la genèse de SuperCollider, voir [12].

[7] « *[An] environment and programming language for real time audio synthesis and algorithmic composition.* » [17] Notre traduction.

[8] Depuis 2001. SuperCollider est actuellement publié sous licence GNU GPL.

[9] « *While synthesis is running, new modules can be created, destroyed, and re-patched, and sample buffers can be created and reallocated. Effects processes can be created and patched into a signal flow dynamically […].* » [13], p. 64. Notre traduction.

[10] Comme la bibliothèque *Just-in-Time Programming* de Julian Rohrhuber et Alberto de Campo [16]. Sur la pratique générale du *live coding*, voir [2].

[11] Le projet approchant le plus cette intention est *BabaKoto* d'Andrea Valle [18]. S'il fournit un ensemble de classes offrant une approche modulaire supportée par une interface graphique efficace, chaque module est équivalent à une instance unique de la classe Synth, ce qui, comme on le verra dans la suite de cet article, est restrictif vis-à-vis de notre objectif tendant à une plus grande généralité.

[12] Héritée des programmes Music n de Max Mathews [10], l'expression *unit generator*, désignant une unité de génération, de lecture, d'enregistrement ou de traitement de signal, est reprise dans plusieurs environnements d'informatique musicale, dont SuperCollider avec la classe UGen.

[13] Le programme décrit dans cet article repose sur la plus récente version stable pour Mac OSX, Linux ou Windows au 29 février 2012 : SuperCollider 3.4.4 (publication en juillet 2011).

[14] Pour une description complète de l'implémentation du serveur audio de SuperCollider, voir [1].

[15] *OpenSoundControl*.

paraphrasée en prose OSC par l'interpréteur sclang, elle-même envoyée au serveur [scsynth]. »[16]

Les *unit generators* de SuperCollider sont représentés au sein de fonctions (*UGen graphs*) passées en arguments d'instances de la classe SynthDef. La figure 1 montre l'implémentation d'un SynthDef décrivant deux sinusoïdes modulées en anneau.

```
SynthDef.new(\ringmod, { arg freq1, freq2;
	var oscil1, oscil2, output;
	oscil1 = SinOsc.ar(freq1);
	oscil2 = SinOsc.ar(freq2);
	output = oscil1 * oscil2;
	Out.ar(0, output);
}).add;
```

Figure 1. SynthDef « ringmod », deux sinusoïdes et un modulateur en anneau

Lorsque le code de la figure 1 est exécuté, le graphe d'*unit generators* du SynthDef est compilé et envoyé au serveur. Il peut alors servir de référence à une ou plusieurs instances de la classe Synth, qui commandent au serveur l'allocation d'un *node* pour le calcul audionumérique et la production sonore correspondante (figure 2).

```
// Création de trois nodes et déclenchement de la
production sonore correspondant au SynthDef
« ringmod »
x = Synth.new(\ringmod, [\freq1, 200, \freq2, 800]);
y = Synth.new(\ringmod, [\freq1, 500, \freq2, 1500]);
z = Synth.new(\ringmod, [\freq1, 100, \freq2, 1000]);

// Destruction des trois nodes
x.free; y.free; z.free;
```

Figure 2. Trois instances de Synth se référant au SynthDef « ringmod »

2.2. Allocation de *buffers*

Certains *unit generators* peuvent faire appel à des mémoires d'échantillons. Celles-ci, représentées par la classe Buffer, sont allouées indépendamment de la déclaration du SynthDef, et sont passées en arguments du Synth correspondant (figure 3).

```
// Allocation de deux échantillons dans deux objets
Buffer
s = Server.default;
a = Buffer.read(s, "échantillon1");
b = Buffer.read(s, "échantillon2");

// Déclaration du SynthDef
SynthDef.new(\lecteur, { arg bufnum;
	var output;
	output = PlayBuf.ar(1, bufnum,
BufRateScale.kr(bufnum));
	Out.ar(0, output);
}).add;

// Création des objets Synth
	// Lecture du premier échantillon
	y = Synth.new(\lecteur, [\bufnum, a]);
	// Lecture du second échantillon
	z = Synth.new(\lecteur, [\bufnum, b]);

// Destruction des objets Synth et Buffer
y.free; z.free; a.free; b.free;
```

Figure 3. Lecteur d'échantillon simple : objets SynthDef, Buffer et Synth

2.3. *Nodes* statiques et dynamiques

Dans les deux exemples précédents, le modulateur en anneau et le lecteur d'échantillon, les *nodes* ont un point commun : ils sont instanciés et libérés par commandes explicites (respectivement les messages « new » et « free » passés aux objets Synth). L'une des caractéristiques de SuperCollider ayant fortement contribué à motiver notre projet est la possibilité de se référer à un SynthDef par la création non d'un objet Synth, mais d'un objet Pbind, qui permet d'associer des motifs (*patterns*) de durées à des motifs de valeurs pour les arguments du SynthDef. Considérons le SynthDef déclaré à la figure 4, représentant une sinusoïde à laquelle est appliquée une enveloppe triangulaire :

```
SynthDef.new(\sinusgrain, { arg freq, amp;
	var envelope, output;
	envelope = Env.new([0, 1, 0], [0.1, 0.1]);
	envelope = EnvGen.kr(envelope, gate: 1,
doneAction: 2);
	output = SinOsc.ar(freq, mul: envelope*amp);
	Out.ar(0, output);
}).add;
```

Figure 4. SynthDef « sinusgrain », sinusoïde et enveloppe triangulaire

L'objet Pbind, à la figure 5, permet de décrire une séquence d'événements (*events*) appelant l'instrument défini par le SynthDef « sinusgrain », et dont les valeurs paramétriques respectent les *patterns* correspondant à ses arguments (fréquence et amplitude). Envoyer le message « play » à cet objet Pbind génère, à partir de grains sinusoïdaux[17], une séquence dont les six premières mesures sont données en représentations musicales solfégiques à la figure 6.

```
p = Pbind(\instrument, \sinusgrain,
	\dur, Pseq([0.5, 0.25, 0.25], inf),
	\freq, Pseq([60, 62, 63, 65].midicps, inf),
	\amp, Pseq([0.5, 0.1, 0.1, 0.5, 0.1], inf)
);

q = p.play;
q.stop;
```

Figure 5. Pbind décrivant une séquence d'événements créant des *nodes* dont la référence est le SynthDef « sinusgrain »

[16] « *Typically, the interpreter translates the code in SuperCollider language in OSC messages for the server. The user writes poetry (so to speak) in the SuperCollider language which is then paraphrased in OSC prose by the sclang interpreter, to be sent to the server.* » [5] Notre traduction.

[17] Grâce à l'argument doneAction de valeur 2 passé à l'objet EnvGen dans le SynthDef « sinusgrain » (figure 4), les *nodes* instanciés par le serveur à chaque nouvel événement de la séquence s'autodétruisent dès que l'enveloppe a terminé sa course, évitant ainsi une accumulation de modules audio et une surcharge du serveur.

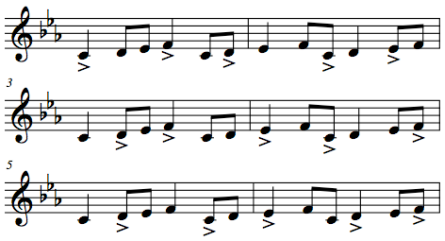


Figure 6. Début de la séquence solfégique correspondant à la description du Pbind de la figure 5

Selon James Harkins, « les *patterns* sont parmi les éléments les plus puissants du langage SuperCollider »[18]. En effet, ils sont associés à un vaste ensemble de classes permettant de composer des combinaisons de séquences complexes, formulées explicitement ou construites algorithmiquement, et de les appliquer à tous les paramètres des unités audionumériques disponibles. Intégrés à l'approche modulaire que nous présentons, ils autorisent également des interconnexions dynamiques : de la même manière que l'exemple représenté aux figures 5 et 6 met en œuvre des séquences affectant les paramètres de durées, de fréquences et d'amplitudes, il est possible d'écrire des *patterns* pour les appliquer aux entrées-sorties d'un module donné, ou aux appels de *buffers* déclarés indépendamment. Dans le cadre de notre programme, nous distinguons deux grandes catégories de *nodes*, tous deux référencés par la classe SynthDef : les *nodes* statiques, qui sont explicitement créés et détruits et sont permanents entre ces deux opérations, et les *nodes* dynamiques, dont l'allocation repose sur le recours à des *patterns* et pour lesquels la structure du SynthDef permet une destruction automatique, par exemple lorsque l'enveloppe d'amplitude atteint son dernier point chronologique (*self-freeing nodes*).

2.4. Bus et interconnexions d'unités

Les classes In et Out permettent à un *node* de recevoir et d'envoyer des signaux en communiquant avec d'autres unités ou avec des interfaces matérielles d'entrée-sortie par l'intermédiaire de bus. La figure 7 montre la création d'un bus par lequel est véhiculé le signal d'une sinusoïde vers le paramètre de fréquence d'une autre sinusoïde.

```
SynthDef.new(\sinus, { arg out, freq;
	var output;
	output = SinOsc.ar(freq);
	Out.ar(out, output);
}).add;

SynthDef.new(\fm, { arg in, freq, modindex;
	var input, output;
	input = In.ar(in, 1); // entrée 1 canal
	output = SinOsc.ar(freq+(input*modindex));
	Out.ar(0, output);
}).add;
```

[18] « *Patterns are one of the most powerful elements of the SuperCollider language* ». [8] Notre traduction.

```
// Allocation d'un bus audio 1 canal
s = Server.default;
a = Bus.audio(s, 1);

// Création et connexion des objets Synth
y = Synth.new(\sinus, [\out, a, \freq, 300]);
z = Synth.after(y, \fm, [\in, a, \freq, 200,
\modindex, 100]);
```

Figure 7. Fréquence de modulation et porteuse connectées par un objet Bus (variable a)

En plus de l'allocation d'un bus, la connexion de plusieurs objets Synth doit être opérée avec une attention particulière à l'ordre d'exécution des *nodes* sur le serveur. Tim Blechmann explique que

> « lorsque des objets Synth sont créés dans scsynth, ils sont ajoutés à une position donnée dans le graphe des *nodes*. SuperCollider dispose de groupes, qui sont des listes concaténées de *nodes*, chaque *node* pouvant être soit un Synth, soit un autre groupe [objet de la classe Group] ; ainsi est établie une structure de données en arbre, avec un groupe pour racine. La position [d'un *node*] peut être spécifiée avec un *node* de référence [dans notre exemple, y] et un argument de relation [ici, "after"], qui peut être "avant" ou "après" le *node* de référence, ou "en tête" ou "à la queue" d'un groupe de référence. »[19]

À partir des quelques notions fondamentales de SuperCollider introduites à travers cette première partie, il est visible que de larges configurations d'unités statiques ou dynamiques, faisant ou non appel à des *buffers*, sont aisées à établir sous forme de code. Pourtant, si la création d'une configuration spécifique, comme dans le cadre d'une composition, se conçoit naturellement par l'écriture explicite de tous les éléments à prendre en considération (*nodes*, bus, *buffers*, *patterns*, ordre d'exécution), il semble fastidieux de disposer d'un certain nombre de modules audionumériques préalablement déclarés et de devoir coder chaque nouvelle interconnexion. La figure 8 montre une situation initiale dans laquelle quatre modules quelconques sont connectés en série entre l'entrée et la sortie physiques de l'ordinateur hôte. Pour insérer un cinquième module dans le trajet de signal, trois opérations doivent être explicitement formulées : la création d'un bus supplémentaire, la création du Synth correspondant au module, et la modification du bus d'entrée pour l'un des modules existants.

```
// Situation initiale
var hardwareinput = 2, hardwareoutput = 0; // Index
semi-arbitraires pour les entrées-sorties physiques
a = Bus.audio(s, 1); b = Bus.audio(s, 1);
c = Bus.audio(s, 1);
```

[19] « *When synths are created on the SuperCollider server, they are added at a certain position in the node graph. SuperCollider has the notion of groups, which are linked lists of nodes, where each node member can be either a synth or another group, effectively modeling a tree data structure with a group as its root. The position can be specified with one reference node and one relational argument, which can be 'before' or 'after' a reference node or at 'head' or 'tail' of a reference group.* » [3], p. 10. Notre traduction.

```
w = Synth.new(\module1, [\in, hardwareinput,
\out, a]);
x = Synth.after(w, \module2, [\in, a, \out, b]);
y = Synth.after(x, \module3, [\in, b, \out, c]);
z = Synth.after(z, \module4,
[\in, c, \out, hardwareoutput]);

// Insertion d'un nouveau module entre x et y
d = Bus.audio(s, 1);
v = Synth.after(x, \module5, [\in, b, \out, d]);
y.set(\in, d);
```

Figure 8. Insertion d'un cinquième module dans une série préalablement établie

Même avec ce contexte et cette manipulation tous deux extrêmement simples, trois opérations doivent être formulées textuellement. Avec une interface graphique objet-cordon comme celles de Max ou PureData, il est envisageable de considérer les opérations suivantes : premièrement, choisir un type de module (« module5 »), deuxièmement, sélectionner le cordon reliant les modules x et y (respectivement « module2 » et « module3 »), troisièmement, choix de l'opération d'insertion. Les créations et affectations de bus sont alors transparentes pour l'utilisateur. Comme on le verra en troisième partie de cet article, de nombreuses manipulations du graphe de modules peuvent appeler un nombre plus important de procédures à exécuter, alourdissant les opérations de codage si une approche textuelle est retenue, mais n'occasionnant qu'un surcoût minime ou inexistant pour l'action à même une interface graphique simple. Nous présentons un premier prototype de celle-ci dans la partie suivante.

3. INTERFACE GRAPHIQUE MODULAIRE EN MAX/MSP/JITTER

Bien que « SuperCollider fournisse différents ensembles de fonctions d'interface graphique utilisateur, ainsi qu'une syntaxe pour écrire de manière transparente du code indépendant de la plateforme »[20], notre choix pour programmer un environnement graphique de manipulation de modules s'est orienté sur Max pour deux raisons. D'une part, la bibliothèque d'objets Jitter comprend des fonctions OpenGL permettant de développer simplement une représentation du graphe de modules selon des cubes reliés par des cordons. D'autre part, les objets Max de la famille *pattr* permettent une gestion efficace des *presets* de paramètres pour les modules individuels.

3.1. Représentation du graphe synoptique de modules en OpenGL

Dans notre prototype actuel, le graphe modulaire est représenté de manière synoptique. À chaque unité modulaire correspond un cube dont la couleur indique le type. L'ordre de disposition des cubes correspond à l'ordre d'exécution des unités dans scsynth, de gauche à droite puis de haut en bas (tableau 1).

0	1	2
3	4	5
6	7	8

Tableau 1. Ordre relatif d'exécution des modules dans scsynth en fonction de leur répartition sur le plan de l'interface graphique (0 : tête de groupe, 8 : queue de groupe).

La figure 9 montre un exemple de graphe reliant six modules de quatre types différents : trois oscillateurs pouvant recevoir un signal modulant en entrée, un lecteur d'échantillon, un mélangeur, un module de transmission vers le convertisseur digital-analogique. La sortie du premier oscillateur est dirigée à la fois vers l'entrée du deuxième oscillateur et vers sa propre entrée.

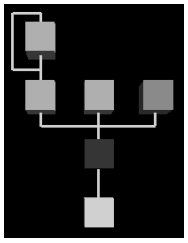


Figure 9. Graphe de six modules : oscillateurs, lecteur d'échantillon, mélangeur, et sortie

Cette représentation est réalisée, à partir de commandes OpenGL envoyées à un objet jit.gl.sketch, sur une fenêtre jit.window : ce dernier objet permet de connaître les coordonnées du curseur de souris et de construire les commandes correspondant aux actions appropriées, comme modifier la couleur des cubes ou des cordons sélectionnés, créer, déplacer ou supprimer un ou plusieurs objets.

3.2. Panneaux des types de modules et des paramètres d'unité

En plus de la fenêtre du graphe général apparaissent toujours au moins trois autres panneaux. L'un permet de sélectionner un type de module (figure 10), le type sélectionné définissant celui de la prochaine unité créée. L'autre est le panneau des paramètres du type correspondant (figure 11). Ainsi, lorsqu'une unité est créée sur le graphe, son type et ses valeurs de paramètres sont déjà déterminés. Le panneau des paramètres étant identique pour plusieurs unités d'un même type, un numéro d'identifiant permet de connaître l'unité à laquelle il est attaché. Enfin, un troisième panneau permet à l'utilisateur de basculer entre deux modes de construction graphique du graphe. Dans le premier mode, chaque nouvelle action sur l'interface commande immédiatement le calcul de modification du graphe audio dans scsynth. Dans le second mode, l'utilisateur opère toutes les modifications graphiques qu'il souhaite sans effet immédiat, avant de déclencher manuellement la mise à jour audio (figure 12).

[20] « *SuperCollider provides for using different gui kits, and also provides syntax for transparently writing kit and platform independent code.* » [7] Notre traduction.

Figure 10. Choix des types de modules et type sélectionné[21]

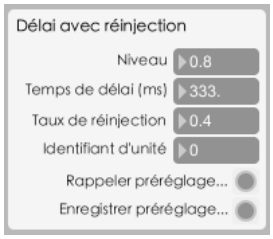


Figure 11. Panneau des paramètres du type « Délai avec réinjection »

Figure 12. Panneau de choix du mode de mise à jour du graphe audio

Dans le premier mode, les modifications sont audibles au fur et à mesure des actions simples (création, déplacement ou suppression d'une unité ou d'un cordon unique). Le second mode permet d'effectuer des transitions directes entre deux configurations modulaires très différenciées (création, déplacement et suppression par lots importants d'unités et de connections).

3.3. Actions de l'utilisateur

Dans l'état actuel du prototype, les actions implémentées se font à partir de la combinaison classique trackpad-clavier ou souris-clavier[22] :
- clic (cube) : sélectionner une unité ;
- clic (cordon) : sélectionner une connexion ;
- clic (vide) : tout désélectionner ;
- cmd+clic (cube ou cordon) : sélectionner une unité ou un cordon sans désactiver les sélections existantes ;
- backspace : supprimer les unités et liaisons sélectionnées ;
- double-clic (espace vide) : créer une unité ;
- clic+glisser-déposer (cube vers espace vide) : déplacer une unité ;
- shift+clic+glisser-déposer (cube vers cube) : connecter deux unités.

Ces actions basiques permettent de modifier le graphe selon les opérations simples implémentées dans SuperCollider : création, déplacement, suppression des unités et des connexions. Les développements futurs de notre interface seront l'objet d'une recherche approfondie sur les possibilités ergonomiques d'actions sur des ensembles d'objets, ainsi que sur des interfaces d'accès gestuel plus appropriées que le binôme clavier-souris.

La représentation du graphe étant synoptique, les cordons n'informent que de l'existence d'une liaison entre une unité et une autre, sans précision sur les index d'entrées-sorties. Lorsque l'action utilisateur consiste à relier un module à un autre et que ceux-ci disposent de plusieurs entrées ou sorties, un panneau permet d'afficher l'index d'entrée ou de sortie visée[23]. À l'aide des flèches gauche et droite du clavier alphanumérique, l'utilisateur peut commander la connexion aux autres entrées-sorties des modules concernés (figure 13).

Figure 13. Panneau de choix des index d'entrées-sorties pour une nouvelle liaison

À terme, l'environnement Max devra permettre un ensemble d'actions spécifiables non graphiquement. Actuellement, une seule fonction existe : générer un graphe d'unités et d'interconnexions aléatoirement[24] (figure 14).

3.4. Sauvegarde et rappel des configurations modulaires

Afin de ne pas devoir reconstruire manuellement un graphe donné depuis une fenêtre vide, un dispositif reposant sur des objets *coll* permet de sauvegarder et de rappeler des configurations modulaires sous forme de fichiers texte[25]. Par ailleurs, pour chaque type d'unité, le panneau des paramètres comprend des boutons de sauvegarde et de rappel de préréglages, qui ouvrent une boîte de dialogue donnant accès aux fonctions avancées d'objets *pattrstorage*[26].

[21] Les modules présentés ici sont donnés à titre de démonstration. Nous exposons en troisième partie les bases de données permettant d'étendre les types de modules disponibles.

[22] Les commandes listées ici sont appliquées à Mac OSX, mais sont généralisables aux autres systèmes d'exploitation.

[23] Par défaut, il s'agit de l'index 0 (première entrée ou première sortie).

[24] À l'origine, cette fonction a été implémentée à des fins de tests. Toutefois, elle nous paraît pouvoir être retenue pour l'expérimentation musicale, et appelle une réflexion sur la mise en œuvre d'autres opérations à caractère aléatoire ou algorithmique.

[25] Au format .txt.

[26] Les informations de préréglages sont importées et exportées au format .xml.

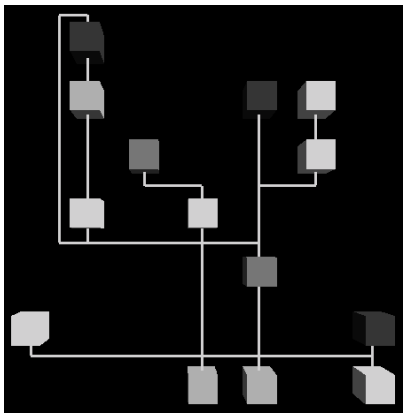


Figure 14. Un graphe généré aléatoirement, sans action utilisateur à même l'interface graphique

Lorsqu'une modification est opérée dans l'environnement Max, aussi bien pour la modification du graphe modulaire que pour l'ajustement paramétrique des unités, les informations adéquates sont transmises à SuperCollider par le biais d'objets udpsend (dans Max) et OSCresponder (dans SuperCollider). Dans la quatrième partie de cet article, nous présentons les opérations par lesquelles SuperCollider adapte son graphe audio en fonction des commandes reçues depuis l'interface graphique.

4. MISE EN CORRESPONDANCE DE L'INTERFACE GRAPHIQUE ET DE SCSYNTH

Dans un article présentant un environnement de simulation d'un synthétiseur modulaire implémenté dans Csound, Eric Lyon décrit son projet en ces termes :

> « La méthode de base de la synthèse analogique consiste à interconnecter des modules autonomes qui génèrent ou modifient des signaux électroniques, pour établir des configurations ayant des qualités sonores particulières. Ce paradigme est adopté dans Csound, où les modules sonores sont remplacés par des *unit generators*, et les cordons de raccordement par des variables de signal. Ce projet a deux parties. D'abord, nous écrirons un langage très simple qui traduit les spécifications de *patch* en orchestres Csound. Ensuite, nous écrirons un programme qui génère algorithmiquement les spécifications de *patch*, automatisant le processus de design sonore. »[27]

En nous inspirant de cette démarche, nous avons écrit dans SuperCollider un programme s'articulant en deux grandes parties. La première contient les bases de données renseignant sur les caractéristiques des unités à intégrer dans notre environnement modulaire. La seconde consiste en un algorithme devant, en fonction des informations reçues depuis Max, modifier le graphe audio généré par le serveur scsynth de SuperCollider.

4.1. Bases de données permanentes dans SuperCollider

4.1.1. Déclaration des objets SynthDef

La programme écrit dans SuperCollider commence par la déclaration des objets SynthDef et leur envoi au serveur, sous la même forme que dans les exemples décrits dans la première partie de cet article. Les instances de SynthDef sont réparties en trois grandes catégories. La première catégorie est celle des objets SynthDef principaux : il s'agit des ensembles d'*unit generators* que l'utilisateur souhaite employer directement à des fins musicales (générateurs, traitements, échantillonneurs). La deuxième catégorie est celle des objets SynthDef de connexion. Comme il le sera précisé au paragraphe 4.2, les connexions entre unités ne peuvent pas toujours être opérées directement : par exemple, envoyer un signal depuis une sortie mono vers une entrée stéréo suppose une conversion, effectuée par un objet Synth. Enfin, la troisième catégorie est celle des objets SynthDef de *bypass*[28]. Pour un module donné, il importe en effet de décrire explicitement le trajet du signal reliant les entrées et les sorties lorsque le traitement est contourné[29]. Pour celles des unités susceptibles d'être désactivées sans être détruites, un SynthDef de *bypass* doit donc être déclaré.

4.1.2. Données relatives aux objets SynthDef

La déclaration des objets SynthDef est suivie de la déclaration d'informations leur étant relatives et nécessaires au fonctionnement de l'ensemble du programme. Pour les SynthDef principaux, ces informations sont les suivantes : nom du SynthDef, statut statique (*false*) ou dynamique (*true*), nombre de *buffers* devant être référencés par le Synth, caractéristiques des entrées-sorties (taux audio ou contrôle[30] et nombre de canaux), nom des arguments (figure 15).

[27] « *The basic method of analog synthesis is to patch together self-contained modules that generate or modify electronic signals into configurations with particular sound qualities. This paradigm is adopted in Csound where sound modules are replaced with unit generators, and patch cords are replaced by signal variables. This project has two parts. First we will write a very simple language that translates patch specifications into Csound orchestras. Second, we will write a program that algorithmically generates patch specifications, automating the sound design process.* » [9], p. 629. Notre traduction.

[28] Nous employons le terme anglais, plutôt que sa traduction « contournement », pratiquement jamais rencontrée.

[29] Par exemple, pour un traitement disposant de deux entrées et de quatre sorties, il peut être souhaitable, en mode *bypass*, de router les deux entrées vers les deux premières sorties, les deux autres sorties ne délivrant alors aucun signal.

[30] Comme le précise Joshua Parmenter, « SuperCollider calcule la sortie [d'*unit generators*] par blocs de valeurs (les échantillons) aussi bien pour les signaux à taux audio (*audio rate : ar*) que pour les signaux à taux contrôle (*control rate : kr*) […]. La taille de bloc par défaut est 64 échantillons, avec un UGen à taux audio calculant 64 échantillons et un UGen à taux contrôle retournant une seul valeur pour chaque bloc. » (« *SuperCollider computes this output in blocks of values (called samples) for both audio rates (ar) and control rate (kr) signals […]. The default block size is 64 samples, with an ar UGen computing 64 samples and a kr UGen returning a single value for each block.* » [14], p. 56. Notre traduction.

```
mainSynthDefData = [
[ \monofm, false, 0,
      [ [[\audio, 1]], [[\audio, 1]] ],
      [\in0, \out0, \freq, \modindex, \amp] ],
[ \stereofm, false, 0,
      [ [[\audio, 2]], [[\audio, 2]] ],
      [\in0, \out0, \freq, \modindex, \amp] ],
[ \eventmonofm, true, 0,
      [ [[\audio, 1]], [[\audio, 1]] ],
      [\in0, \out0, \freq, \modindex, \amp] ],
[ \controlmonofm, false, 0,
      [ [[\control, 1]], [[\control, 1]] ],
      [\in0, \out0, \freq, \modindex, \amp] ],
[ \sampleplayer, true, 1,
      [ [[\audio, 1]], [[\audio, 1]] ],
      [\in0, \out0, \bufnum, \loop, \amp] ]
];
```

Figure 15. Exemple de base de données mainSynthDefData

Les bases de données relatives aux objets SynthDef de connexion et aux objets SynthDef de *bypass* ont pour information le nom de SynthDef et le nom des arguments.

4.1.3. Données relatives aux types d'unités modulaires

À chaque cube de notre interface graphique correspond ce que nous appelons ici une « unité ». Pour une flexibilité optimale, aucun *a priori* n'est donné à la structure d'un type d'unité : elle peut aussi bien correspondre à un SynthDef simple qu'à un ensemble d'objets SynthDef statiques ou dynamiques. En fonction de ses intentions, l'utilisateur peut définir un type d'unité comme étant un oscillateur simple ou une table de mixage virtuelle comprenant des objets SynthDef d'équalisation paramétrique, de compression et d'autres traitements. Avant de déclarer un type donné, il importe de savoir que sa structure interne n'est pas modulable (y compris en termes de nombres de canaux pour les entrées-sorties). Par exemple, si un simulateur de DX7 doit être établi, il conviendra, afin de pouvoir configurer les modulantes et les porteuses selon les différents algorithmes de l'instrument original, de déclarer non pas un type « DX7 », mais un type « oscillateur avec entrée pour modulation », dont on instanciera six unités sur le graphe, selon l'une des trente-deux interconnexions définies par Yamaha. Lorsqu'un type d'unité est défini, l'utilisateur peut, depuis Max, en créer autant d'instances qu'il le souhaite[31]. Les données définissant un type d'unité sont structurées sous la forme d'un *array* d'informations : le graphe des objets SynthDef principaux, définissant les *nodes* à relier et leurs connexions internes à l'unité, les sorties vers l'extérieur de l'unité, le nom du SynthDef de *bypass* à utiliser, l'ensemble des *buffers* utilisés par l'unité, l'ensemble des *patterns* utilisés par l'unité. Dans l'exemple de la figure 16, le premier type est composé d'un seul *node*, ayant pour référence le SynthDef « monofm », dont la base de données mainSynthDefData, à la figure 15, nous informe qu'il est statique (deuxième élément : *false*).

```
userTypeData = [
[ // Premier type
      [ // Graphe des objets SynthDef principaux et
de leurs entrées
            [ \monofm, [[[-1, 0]]] ] ],
      [ [ 0, 0 ] ],// sorties
      \bypass1,     // bypasser
      nil,          // buffers
      nil           // patterns ],
[ // Deuxième type
      [ // Graphe des objets SynthDef principaux
            [ \sampleplayer, [[[-1, 0]]] ] ],
      [ [ 0, 0 ] ],// sorties
      nil,          // bypasser
      buffers[0],   // buffers
      patterns[0]   // patterns ],
[ // Troisième type
      [ // Graphe des objets SynthDef principaux
            [ \monofm, [[[-1, 0]]] ],
            [ \monofm, [[[-1, 1]]] ],
            [ \eventmonofm, [[[0, 0], [1, 0]]] ] ],
      [ [ 2, 0, 0 ], [ 2, 0, 1 ] ], // sorties
      \bypass3,                    // bypasser
      nil,                         // buffers
      patterns[1]                  // patterns ]];
```

Figure 16. Base de données de trois types d'unités

L'*array* [[[-1, 0]]] lui étant associé signifie que sa première entrée correspond à la première entrée de l'unité elle-même. L'*array* suivant (« sorties ») indique que la première sortie de l'unité correspond à la première sortie du premier Synth du graph ([0, 0]). Ensuite, sont indiqués le nom du SynthDef de *bypass* pour l'unité (« bypass1 »), et les deux derniers éléments de l'*array* d'unité indiquent que celle-ci n'utilise aucun *buffer* ni aucun *pattern* (nil, nil).

Le troisième type est composé de deux objets Synth statiques (« monofm ») et d'un *node* dynamique (« eventmonofm »). Les *arrays* d'entrées et de sorties indiquent les connexions suivantes : la première entrée du premier « monosynth » correspond à la première entrée de l'unité ([[[-1, 0]]]), la première entrée du second « monosynth » correspond à la seconde entrée de l'unité ([[[-1, 1]]]). Le SynthDef « eventmonofm » disposant d'une seule entrée, mais étant dynamique, cette entrée sera, en fonction de la séquence du *pattern* utilisé, liée à la première sortie du premier « monosynth » ([0, 0]) ou à la première sortie du second « monosynth » ([1, 0]). De la même manière, en fonction de la même séquence, sa première et unique sortie sera envoyée à la première sortie de l'unité ([2, 0, 0]) ou à la seconde sortie de l'unité ([2, 0, 1]). La figure 17 récapitule la structure d'une unité de type 3.

À partir des bases de données relatives aux objets SynthDef (mainSynthDefData) et aux types d'unités définies par l'utilisateur (userTypeData), est développée une autre base de données (typeData), qui collecte toutes les informations nécessaires à la bonne gestion du graphe audio lors des manipulations graphiques.

[31] Dans la limite des ressources de l'ordinateur hôte.

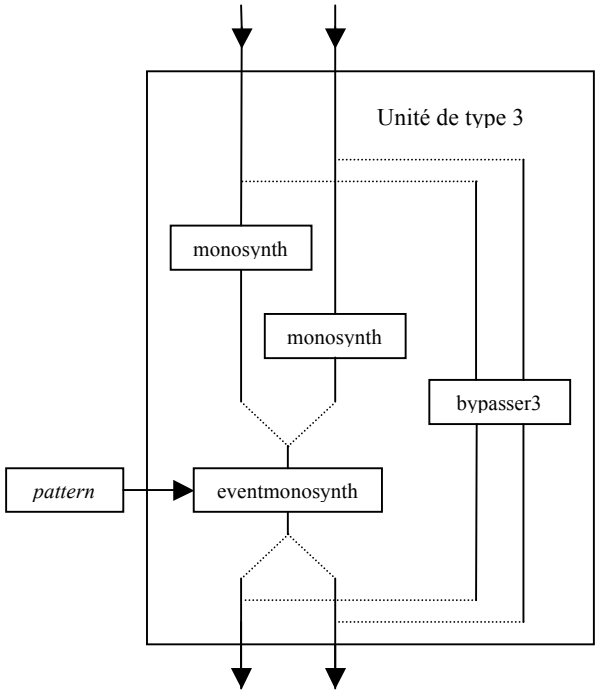


Figure 17. Structure d'une unité de type 3.

4.2. Algorithme de construction et de gestion du graphe audio

Les informations liées à chaque unité créée (objets Group, Synth, Bus, Buffer, *patterns* et informations diverses) sont regroupées dans une base de données dynamique, qui permet d'identifier chaque unité selon son index d'emplacement, inchangé depuis sa création jusqu'à sa destruction. Lorsque des actions sont opérées par l'utilisateur sur l'interface graphique, Max envoie à SuperCollider un message formaté selon l'*array* représenté à la figure 18. À partir de ces informations relatives aux suppressions, créations et déplacements d'unités et de connexions, l'algorithme que nous avons implémenté calcule l'ensemble des opérations de reconfiguration en les exécutant dans l'ordre suivant :
- création des groupes et des structures de données pour les nouvelles unités ;
- création éventuelle des *buffers* et *patterns* associés aux nouvelles unités ;
- création des nouveaux bus et allocation de leurs index à une structure de gestion des bus ;
- création des objets Synth (référençant les objets SynthDef de connexion, de *bypass* et principaux) ;
- modification des positions d'exécution et des indices de bus pour les *nodes* appropriés ;
- suppression des groupes, objets Synth, bus, *buffers* et *patterns* devenus inutiles.

À partir des informations de la base de données « typeData » et des informations de modification du graphe, les calculs relatifs à la gestion des objets Synth et des bus opèrent selon la logique suivante. Soient S la sortie d'un module A et E l'entrée d'un autre module B ou du même module, et l'établissement d'une nouvelle connexion de S à E.

```
modifGraphArray = [
[ // Opérations sur les unités
      [ 1 ], // Suppression de l'unité 1
      [[ \tail, [2, 0]]], // Création d'unités de
type 2 et 0 en queue de graphe
      [[ 0, [4, 5]]] // Déplacement des unités 4 et 5
avant l'unité 0 ],
[ // Opérations sur les connexions
      [ ], // Pas de suppression explicite de
connexion
      [[ [ 4, 0 ], [[ 0, 0 ]] ]], // Création d'une
connexion entre la première sortie de l'unité 4 et la
première entrée de l'unité 0
      [ ] // Pas de déplacement de connexion ]];
```

Figure 18. Informations communiquées depuis Max pour la modification du graphe audio de scsynth

Si S délivre un signal au taux audio et si B précède A dans l'ordre d'exécution du graphe audio, la connexion implique la création d'un module de *feedbacker* entre les deux unités[32]. Si S délivre un signal au taux audio et E doit recevoir un signal au taux contrôle ou inversement, la connexion implique la création d'un module de conversion de taux de signal. Si S délivre un signal circulant sur un nombre de canaux différent de celui pouvant être reçu par E, la connexion implique la création d'un module de conversion du nombre de canaux (par mélange ou par copie). Enfin, si plusieurs sorties sont dirigées vers une même entrée, la création d'un module de mixeur est effectuée. Grâce à l'emploi de ces objets Synth, référencés par les objets SynthDef de connexion, l'utilisateur peut, de manière entièrement transparente, décider de n'importe quelle interconnexion, sans limitation imposée par les caractéristiques de connectique des modules.

5. CONCLUSION

Sous sa forme actuelle, le programme dont nous avons présenté les principaux aspects permet une manipulation simple, flexible et intuitive des *unit generators* de SuperCollider et de leurs possibilités de contrôle dynamique. Une fois la détermination d'unités articulant des objets SynthDef effectuée, l'utilisateur peut en explorer les possibilités musicales et sonores en les interconnectant selon n'importe quelle configuration, d'après le modèle des instruments modulaires rencontrés parmi les lutheries analogiques et numériques. La gestion des différents éléments inhérents à l'implémentation d'un graphe donné (objets Group, Synth, Bus notamment) étant rendue transparente, un important travail de codage, redondant pour qui souhaite utiliser des modules identiques selon différentes combinaisons, est ainsi évité.

L'intérêt de ce programme est potentiellement important pour une large variété d'utilisateurs : utilisable dans le temps de la composition aussi bien qu'en tant qu'instrument de performance temps réel, il constitue un outil efficace et facilement extensible. Pour

[32] Avec la classe InFeedback.

le programmeur averti comme pour le musicien désireux de découvrir les ressources de SuperCollider, l'intégration de types de modules est facilement accessible et peu coûteuse en temps de travail : la déclaration de nouveaux SynthDef et la mise à jour des bases de données de SynthDef et de types d'unités permet la manipulation immédiate de ces derniers au sein de l'environnement existant.

D'ores et déjà exploitable et fonctionnel, le programme comporte de nombreuses limites qui constituent autant de perspectives pour ses développements futurs. En premier lieu, une gestion efficace des *buffers* et des *patterns* devra être implémentée dans l'interface Max. La création d'une mémoire ou d'un motif est encore attachée de manière trop rigide à la création de l'unité correspondante[33] : dans la mesure où ces éléments sont dissociés dans SuperCollider, il convient de permettre à l'utilisateur de gérer sa propre banque d'échantillons et de séquences indépendamment des modules manipulés. Ensuite, la déclaration des types d'unités peut être encore facilitée, par l'implémentation d'un dispositif d'accès graphique pour la création d'un nouveau module. En ce sens, les travaux sur *InkSplorer* présentés par Jérémie Garcia, Theophanis Tsandilas, Wendy E. Mackay et Carlos Agon aux Journées d'Informatique Musicale de 2011 [6] sont d'un grand intérêt pour définir des modules par représentation directement sur papier des unités, entrées-sorties et connexions internes. Enfin, si l'interface synoptique du graphe implémentée dans Max permet une bonne visualisation et favorise des actions simples à partir du binôme trackpad-souris, ses développements devront prendre en compte les possibles interfaçages pour l'utilisateur. Parmi les perspectives que soulève notre programme, son optimisation ergonomique appelle une recherche véritable sur les accès gestuels appropriés à une création musicale reposant sur la modularité virtuelle.

6. REFERENCES

[1] Bencina, R. « Inside scsynth », in Wilson, S., Cottle, D. et Collins, N. (éd.), *The SuperCollider Book*, The MIT Press, Cambridge (MA), 2011, p. 721-740.

[2] Blackwell, A. et Collins, N. « The Programming Language as a Musical Instrument », *Proceedings of the 17th Workshop of the Psychology of Programming Interest Group*, Brighton (UK), Univ. of Sussex, 28 juin-1er juillet 2005, p. 120-130.

[3] Blechmann, T. *Supernova – A Multiprocessor Aware Real-Time Audio Synthesis Engine For SuperCollider*, mémoire de Master en informatique musicale, dir. Ertl, A., Vienna Univ. of Technology, Vienne, 2011.

[4] Bossis, B. « Écriture instrumentale, écriture de l'instrument », in Stévance, S. (dir.), *Composer au XXIe siècle. Pratiques, philosophies, langages et analyses*, Vrin, coll. Musicologies, Paris, p. 119-135.

[5] *Client versus Server Architecture and Operations*, document d'aide intégré à SuperCollider 3.3.4

[6] Garcia, J. Tsandilas, T., Mackay, W. E. et Agon, C. « *InkSplorer* : vers le papier interactif pour la composition musicale », Actes des Journées d'Informatique Musicale 2011, Saint-Étienne, Univ. Jean Monnet, 25-27 mai 2011, p. 237-238.

[7] *GUI Overview and Introduction*, document d'aide intégré à SuperCollider 3.3.4

[8] Harkins, J. « A Practical Guide to Patterns », document d'aide intégré à SuperCollider 3.3.4

[9] Lyon, E. « A Modular Synthesizer Simulation Program », in Boulanger, R. et Lazzarini, V. (éd.), *The Audio Programming Book*, The MIT Press, Cambridge (MA), 2011, p. 629-653.

[10] Mathews, M. *The Technology of Computer Music*, The MIT Press, Cambridge (MA), 1974.

[11] *Max / Cycling'74*, http://cycling74.com/products/max/ (lien vérifié le 29 février 2012).

[12] McCartney, J. « Foreword », in Wilson, S., Cottle, D. et Collins, N. (éd.), *The SuperCollider Book*, The MIT Press, Cambridge (MA), 2011, p. IX-XI.

[13] McCartney, J. « Rethinking the Computer Music Language: SuperCollider », *Computer Music Journal*, vol. 26, n° 4, hiver 2002, p. 61-68.

[14] Parmenter, J. « The Unit Generator », in Wilson, S., Cottle, D. et Collins, N. (éd.), *The SuperCollider Book*, The MIT Press, Cambridge (MA), 2011, p. 55-80.

[15] Risset, J.-C. « Évolution des outils de création sonore », in Vinet, H. et Delalande, F. (dir.), *Interfaces homme-machine et création musicale*, Hermes Science Publications, Paris, 1999, p. 17-36.

[16] Rohrhuber, J. et De Campo, A. « Just-in-Time Programming », in Wilson, S., Cottle, D. et Collins, N. (éd.), *The SuperCollider Book*, The MIT Press, Cambridge (MA), 2011, p. 207-236.

[17] *SuperCollider / About*, http://supercollider.sourceforge.net/ (lien vérifié le 29 février 2012).

[18] Valle, A. « BabaKoto », http://www.fonurgia.unito.it/andrea/wikka.php?wakka=BabaKoto (lien vérifié le 29 février 2012).

[19] Wanderley, M. et Depalle, P. « Contrôle gestuel de la synthèse sonore », in Vinet, H. et Delalande, F. (dir.), *Interfaces homme-machine et création musicale*, Hermes Science Publications, Paris, 1999, p. 145-163.

[33] Par exemple, la création d'une unité de type « échantillonneur » entraîne automatiquement l'allocation d'un nombre fixe d'objets Buffer. La création d'une unité de granulateur d'un type donné entraîne invariablement la création d'un même *pattern*.

A LAZY REAL-TIME SYSTEM ARCHITECTURE FOR INTERACTIVE MUSIC

David Janin
Université de Bordeaux, LaBRI UMR 5800,
351, cours de la libération,
F-33405 Talence
janin@labri.fr

ABSTRACT

Designing systems that function both in real-time and are interactive, characteristics commonly encountered in computational music, is a challenging task indeed. It becomes even more difficult if we require these systems to be generic with respect to the underlying interactive scores that are to be followed.

The aim of this paper is to define a generic system architecture of this type. Our proposal is based on a lazy real-time kernel that manages both scheduled synchronous events and unpredictable asynchronous inputs in reactive fashion.

This computation by need approach contrasts with standard real-time architectures where the real time kernel is built upon an active periodic loop. It also allows for a clear distinction to be established between interactive music programs written in symbolic time, and interactive music performance executed in real time.

1. INTRODUCTION

Designing musical systems that function both in real-time and are interactive is a challenging task.

At the lowest level, real-time requirements induce a *synchronous* slicing of time with a period defined by a *fixed time quantum*. Computations are limited, for they need to be performed at a very quick pace. For instance, inputs are audio streams possibly filtered by analyzers while a fixed data-flow diagram produces outputs from inputs [4, 7]. Contrary to this, at the interaction management level, some processes are guarded by the advent of external *asynchronous* events. In that case, these data-flow diagrams can be dynamically restructured when event arrived.

In other words, at the lowest level, interactive music systems synchronously compute *sound values*. At the highest level, interactive music systems asynchronously compute *time structures* upon which music itself is built. Even if systems delegate the actual sound production to subcomponents, there is still a need for low level synchronous management of these delegations.

This distinction between low level real-time computations and high-level interactions also appears in the underlying time scale they are based upon. At the lowest level, the *time quantum* is generally measured in 10^{-5}th of a second, e.g. at a $44kHz$ sampling rate. At the interactional level, the *musical tempo* is measured in 10^{-1}th of a second, e.g. from 30 to 300 beats per minute. In between, the expected *reaction* or *precision time* of the system after an asynchronous event is measured in 10^{-3}th of a second, i.e. the lower limit beneath which standard human perception no longer discerns the difference in beat positions.

This shows that mixing real-time and interactional requirements requires a clear distinction to be made between, on the one hand, high level *interactive music controls* governed by a *symbolic time progression* in the underlying *interactive score* and, on the other hand, low level *music production* based on the ticking of a *real time clock*.

Merging these two levels when designing a system will probably give rise to design Flaws. The obtained software will probably be non modular and non reusable. Still, these two levels of design must function hand in hand. The aim of the system architecture explored in this paper is to provide a clear framework for a partnership of this sort.

Main contribution

In this paper, we aim at proposing an abstract generic system architecture for interactive real-time music performance that will fulfill all of the above requirements. At the system architecture level, the central question we address is how the various components in interactive music software will interact.

Our proposal is based on a lazy real-time kernel that handles, in a reactive way, both scheduled synchronous events and unpredictable asynchronous inputs. This contrasts with standard real-time architecture built upon periodic reactive loops.

The resulting real-time and interactive system architecture is peculiarly robust with respect to occasional time drifts : whenever forced out of time, the running system will auto-stabilize on time as if no time drift had ever occurred.

Within the specific framework of music and the architecture of the system thereby established, we also encode a clear distinction between symbolic time (beats) and real-time (seconds), each being related to the other via a constantly changing tempo.

As a consequence, despite its simplicity, this model induces several layers : from low-level input/output controllers to high-level music controllers, each with its own clear and distinct functional specifications. In particular, the music controller layer may simply be seen as an abstract interpreter of (arbitrary) symbolic interactive scores. This guarantees genericity.

Related works and subject position in the field

The last decades have seen the development of various software programs for Computer Assisted Music either used on stage for live performances or integral to multimedia applications for rich interactive audio supports. These softwares range from low level sound synthesis and control tools such as *Faust* [7] or *Max/MSP* [4], to high level composition assistants such as *Elody* [15] or *OpenMusic* [1] to name but a few.

However, the design and execution of computer assisted interactive music still remains a challenging task. We first need to gain a better understanding of the way low level(synchronous) sound synthesis and control may be combined with high level (asynchronous) musical inputs. It some sense, there is a increasing need for mixed systems that provide high level interactive control structures for the description of potentially complex interactions between lower level sound or music features.

Many systems of this sort, see [5] among others, can be seen as forms of *domain specific languages (DSL)* that are adapted to the design and implementation of interactive scores. Do these languages attain a sufficient level of abstraction ? Do they induce an adequate notion of interactive scores ? The pragmatic reuse of existing and reliable low level tools somehow messes up the picture.

There is as yet no appropriate yardstick for measuring the expressivity of interactive music description. Even more importantly, the temporal and spatial means of structuring interactive scores, thereby aiding composers in this highly arduous task, still need to be better understood and developed.

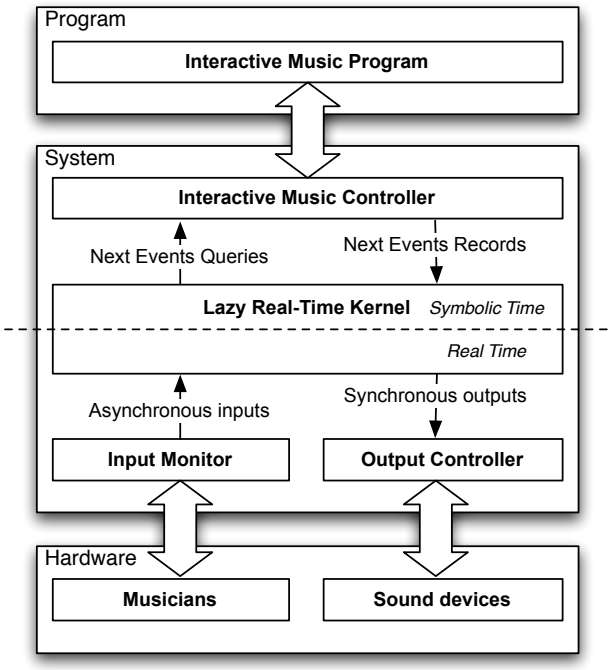
This paper, without forasmuch being able to clearly define the exact *nature* of an interactive musical score, aims at establishing a more precise understanding of *where* and *how* such interactive scores may be played. As a result, we may develop an abstract operational semantics for such scores that has some similarity with Alur and Dill timed automata [3] or, more generally, hybrid systems [11].

This enforces the general and well-accepted idea that known models, developed for years in the presumably distinct application context of critical embedded systems, may nonetheless be adapted efficiently to the application context of interactive music.

2. GENERAL ARCHITECTURE

The main purpose of a system is to bridge the gap between the program layer and a hardware layer. Our proposal, oriented towards interactive music performance, follows this general specification.

Its main components and connection with the environment are depicted in the diagram below.



The system itself is composed of three layers of components whose functionality may be described with more details.

The interactive music controller. This component acts as an interactive partition (or interactive music program) follower ; it receives queries from the musical events in the real-time kernel with symbolic time stamps and produces back to the real-time kernel, in coherence with the score followed and ? ? the record of musical events to be performed at the next relevant symbolic time value.

Lazy real-time kernel. This component handles the lazy real-time loop (described below in detail), a call by need real-time scheduler ; functionally, it handles the communication between the symbolic time layer defined by the music controller and the real time layer defined by both the input monitor and the output controller ;

Input monitor / output controller. These components are in charge of the communication between the interactive music system and the music hardware :

(a) the input subcomponent receives asynchronous input events from musical instrument users (musicians) and transmits their formatted descriptions to the real-time kernel with no delay,

(b) the output subcomponent produces musical streams upon reception of their formatted descriptions received from the real-time kernel.

3. LAZY REAL TIME LOOP

The lazy real time loop, running in the real-time kernel, is essentially in charge of the reactive communication between the music controller denoted M, the input monitor denoted by I and the output controller denoted by O.

It is described below in an object-oriented syntax.

```
Event * E; // eventsList
Time * T;  // next firing date
T = M.getNextEventDate();
While (T.isDefined()) do
{
  Event * E = I.WaitEventUntil(T);
  if (E.isDefined())
     // Received event case
     E = M.updateReceiEvent(E,Now);
  else
     // Planed event case
     E = M.getNextEventAtTime(T);
  O.fireEventAtTime(E);
  T = M.getNextEventDate();
}
```

This loop structure calls for some explanation. As opposed to standard real time architectures, it is built upon a mechanism involving a lazy reactive evaluation schema triggered by *two competing events* :

(1) an *unpredictable asynchronous event* E is received from the input monitor I ; in that case, the event is passed with no delay to the music controller that sends back a possibly enriched event description that is fired immediately ; such an event is called a *received event*,

(2) a *scheduled next event date* expires and the new scheduled synchronous event E, provided by the music controller M, is passed to the output controller to be fired at time T ; such an event is called *scheduled* or *planned events*.

The monitoring of these two competing events is implemented via the call of `WaitEventUntil(T)` on the input monitor I. This method returns the undefined object `nil` when no received event has occurred and the current date is greater [1] or equal to the next event date encoded in T. We describe how these events are managed in further detail below.

Management of a received event. By default, when an unpredictable event E is received, a copycat scenario takes place. This scenario is implemented as follows. Method `updateReceiEvent(E,Now)` sends back the event E. The next event scheduled date, returned by the next call to `getNextEventDate(T)`, remains unchanged.

In all cases, the received event is passed to the music controller which may then, upon reception, update its own control states. The event actually fired, sent back by the controller, may even be different to the received event. It may have been enriched. It may even be ignored when the music controller sends it back `nil`.

More precisely, the musical consequence of the reception of an asynchronous unpredictable event, that depends both on its reception date and its value, is governed by the interactive score followed by the music controller. For instance, it may even be the case that the next planned event date changes after updating by the music controller. This generally happens when a received event induces a change of tempo.

Management of a planned event. When a scheduled event date expires, the music controller is asked for the event E to be fired. This is done by the lazy real-time kernel, using the `getNextEventAtTime(T)` method. By default, it simply reads the music score, sends back the next event, and updates its own record of the next scheduled event date.

Until that firing date, there is no need to know which event is to be fired. It follows that this event may be simply *computed* by the music controller, when asked via the `getNextEventAtTime(T)`. This means that the music score can be truly interactive, in the sense that, at any time, the played event may depend on the history of the musical events that have been received and produced so far.

At any time, the next musical event to be played depends on the current internal state of the interactive music score. This programmatic feature of the score is discussed a little further in Section 5 below.

Firing events on time. In all cases, both (enriched) received events or planned events are sent to the output monitor to be fired immediately. In practical implementations of this process, we make the firing of a planned event a little more subtle.

More precisely, method `WaitEventUntil(T)` resumes some `delta` seconds *before* the real time scheduled date expires. This anticipates the amount of time needed for the computation of the next event E. Firing a planned event is thereby performed as follows :

(1) if the current date is sooner than the scheduled firing date, the real-time kernel waits during the remaining lapse of time ; the event is then *fired just on time*, i.e. this is the expected default case,

(2) if the current date is equal or `gamma` seconds later than the scheduled firing date, the event is immediately fired *almost on time*, i.e. `gamma` is the allowed lapse of time for an error in precision,

(3) if the current date is greater than the scheduled firing date plus `gamma`, this means the system is late ; the firing of the event can be omitted in order to avoid parasite noises which are out of time ; it is expected that the system will auto-stabilize.

Parameters `delta` and `gamme` may be set adequately according to the performance of the computer the system is running on.

1 . this may happend when the all system is late

4. REAL TIME VS SYMBOLIC TIME MANAGEMENT

One of the major tasks of the lazy real-time kernel is to ensure the conversion from real time, handled by the input monitor and the output controller, to symbolic time handled by the interactive music controller.

In the lazy real-time loop described above, we have concealed the means by which real-time dates are converted into symbolic dates and vice-versa, by which symbolic dates are converted into real-time dates.

The methods and attributes of class `Time` manage these conversions operating back and forth.

4.1. Real and symbolic current time handling

The first basic attributes of the class `Time` are :

(1) `SymbCurrentD` defined as the *symbolic current date*, i.e. the (float) number of *symbolic time units* (or *beats*) elapsed since the beginning of the musical performance until *now*,

(2) `RealCurrentD` defined as the *real-time current date*, i.e. the (float) number of *real time units* (or *minutes*) elapsed since the beginning of the musical performance until *now*,

(3) `tempo` defined as the evolving speed of the symbolic date w.r.t. the real-time date (in *beats per minute*).

Observe that at any time, the value of the *real-time current date* is defined while the value of the *symbolic current date* needs to be computed.

In the simplest case, when the *tempo* is constant, the following *invariant property* holds.

```
SymbCurrentD == tempo * RealCurrentD;
```

This shows how the *symbolic current date* can be computed from the (constant) value of the *tempo* and the *real-time current date*.

In the general case, when the *tempo* is not constant, things are a little more complex. In this light the following hypothesis may be posited :

> (H) Between any two successive events, be they received or planned events, *tempo* is constant.

May we therefore argue that this hypothesis is a constraint ? We may observe that any change of tempo [2] can be modeled as an event in its own right and that therefore this hypothesis is simply a modeling choice.

On the basis of this simple hypothesis, managing the symbolic vs real-time conversion may be achieved by recording the *last* values of symbolic or real-time dates in two extra attributes. More formally, these extra attributes of the class `Time` are :

(4) `SymbLastD` defined to be the *symbolic last event date*,

(5) `RealLastD` defined to be the *real-time last event date*.

The *invariant property* associated with these new attributes is defined as follows :

```
SymbCurrentD == SymbLastD +
    tempo * (RealCurrentD - RealLastD);
```

This shows, in the general case, how the value of the *symbolic current date* may be computed from the value of the *real-time current date*.

It may be observed that this equation is essentially needed when updating the music controller's current state upon the reception of an event `E`. Indeed, the music controller `M` only handles symbolic time in the interactive music score. Converting the real-time date of reception of a given event `E` into its corresponding symbolic time value is thus a necessity.

4.2. Scheduled dates updates

In the lazy loop described above there is yet another notion of time which needs to be modeled : namely, the *scheduled symbolic date* and *scheduled real-time date* of the next planned event.

This is done by using two more attributes in the class *Time* which are :

(6) `SymbSchedD` defined as the *symbolic scheduled date* of the next planned event,

(7) `RealSchedD` defined as the *real-time scheduled date* of the next planned event.

These scheduled dates are related with last dates in the same way as current dates are related with last dates. However, the *symbolic scheduled date* is *provided* by the music controller when computing the next event date. It follows that we now need to compute the *real-time scheduled date* from that symbolic value.

The relevant *invariant property* is thus defined as follows.

```
// whenever needed
RealSchedD = RealLastD +
     (SymbSchedD - SymbLastD)/tempo;
```

This shows how the value of the *real-time scheduled date* is computed from the value of the *symbolic scheduled date*.

This equation is essentially required when the input monitor `I` is waiting for a input event prior to some scheduled date `T`. Since monitor `I` only handles real-time dates, the symbolic scheduled date provided by the music controller will necessarily have to be converted.

4.3. Last date updates

We are now ready to describe the update procedure of the *real-time last event date* and the *symbolic last event date*. By definition, these dates must be updated every time an event is fired.

2 . either from the input monitor or from the music controller

At first sight, intuition might lead us to intuit that these updates values are to be calculated with the values of the *real-time* and the *symbolic current date* of the given event production. Yet this intuition would indeed be wrong. By definition, the *real-time current date* changes all the time. It may even be the case that *real-time current date* is greater than *real-time scheduled date* since firing an event also takes a certain amount of time. Even worse, it may be the case that the *real-time current date* values changes from the moment we *want* to read its value from the moment we actually *ascertain* its value.

It transpires that these updates must be computed with the values of the *real-time* and *symbolic scheduled dates* of the event that has just been performed. In other words, we actually perform the following update :

```
RealLastD = RealSchedD;
SymbLastD = SymbSchedD;
```

The updating of the tempo, an update that may be associated with the event fired, occurs immediately after the last date updates :

```
tempo = E.newTempo();
```

In order to increase the robustness of the code, the new tempo, associated with any event, may be set, by default, in accordance with the current tempo value.

4.4. Properties defining the robustness of the lazy time handling

This handling of symbolic and real-time dates enjoys a number of key properties which are worthy of discussion.

Time precision. With this architecture, scheduled dates are *computed* from previous scheduled dates and tempo at any moment in the run of an interactive score. It follows that the time precision may be measured, say, just before firing an event, as follows :

```
timePrecision =
       RealCurrentD - RealSchedD;
```

which is positive when the firing of the event occurs *after* the scheduled date.

Experiments on a prototype implementation of that system in *ObjectiveC* under *MacOSX* shows that this time precision just remains below a few ms which is just enough for musical performance.

Robustness w.r.t. time drifting. When firing an event, if the time precision is too great, then we may seek to avoid the sound resulting from this event being produced. It results that this is easily implemented by simply *guarding* the actual firing of an event by a comparison between the measured time precision and the maximal allowed one.

In doing so, the resulting system becomes remarkably robust : if the system is paused for some reason (either intentionally or because of an overload of the computer running the system) then, upon resuming, the system not only omits to play the outdated events, but also, since the scheduled dates are computed data, the system runs forward through the score until it reaches the correct symbolic date corresponding to the actual real-time date.

In other words, after a pause, the system resumes as if *no pause had ever occurred* ! In a live performance context, especially when real musicians continue to play while the system is paused, or when listeners are dancing or even just finger tapping, the fact that the system will resume on time is a particularly desirable property.

We may observe that this property is not satisfied by standard *streaming software* since, quite often, audio or video frames are not time stamped.

5. INTERACTIVE MUSIC SCORES

At this point, the real-time kernel of the proposed system requires further analysis. As the input monitor and output controller are rather simple at that level of abstraction, it remains for us to describe the way the interactive music controller can be *programmed* in greater depth.

To some extent, the interactive music controller is a symbolic execution layer upon which an interactive music specification, no longer seen just as a score to be followed, is run. In our approach : *an interactive score is defined as a timed reactive program that produces the musical score on line, event after event, in step with the history of the received input events.*

In this section, the characteristic of such musical programs will be described in greater depth. It is not our intention to defend a given *syntax* for these programs. We are more concerned by the operational *semantic* features these programs may have.

5.1. Some basic musical programs

In order to better intuit how such timed interactive programs may be defined, we describe below several typical musical scenarios and show how they may be encoded.

Immediate start. The first start scenario envisaged arises when we want the music to be started immediately upon activation of the system.

This can be done by a controller that sets the initial next scheduled event date to zero. Indeed, in doing so, the real time kernel immediately prompts the music controller for the first musical event to be performed.

Conditional start on input. Contrary to this, another possible start scenario arises when we want the music to be started by an external input event that may occur after an unpredictable delay.

In turn, this may simply be achieved by a controller that sets the initial next scheduled event date to infinity ($+\infty$). This way, the system will necessarily wait for an external event.

Observe that if such an infinite date value is not available, this can still be done by repeatedly producing a silent scheduled event until the first external event is received.

End scenario. We may also ask how such a system may be stopped. Might we therefore argue that the architecture described here engenders never ending musical pieces ? Actually, the lazy loop makes this quite clear. The music controller stops the system by simply sending an undefined (or *nil*) next event date. In other words, this undefined date acts as the final bar of an interactive score.

Play through metronome scenario with varying tempo. Finally, a metronome with play through capacity is also easily encoded as a music controller.

Indeed, repeatedly, the symbolic date of the next scheduled event is by default increased at every beat by one : the metronome is expected to tick at every beat. At any other date, upon reception of an external event `E`, the default behavior described in Section 3 is executed, i.e. method `updateReceiEvent(E,Now)` simply sends back the event `E`.

In doing so, a simple additional input interface with a tempo change cursor and a start/stop button may complete the picture in order to produce the missing start/stop and tempo change events.

5.2. Music programs as symbolic timed automata

At any scheduled date, the interactive music controller essentially provides the next scheduled event to be fired. Of course this event must be known before being fired. However, until its firing date, any asynchronous external event may occur and change this characteristic. This means that the next scheduled event must be computed right on time when needed for firing : this computation by need is the consummate definition of lazy computation.

But what about interactive scores ? The proposed architecture permits the programming of timed controllers in an almost pure symbolic time setting. The real time interpretation is solely governed by the evolving variable `tempo`. Indeed, this tempo may be changed either by hand with adhoc input events or programmatically by special control events from the symbolic controller (see the end of Section 4.3).

This means that the interactive music controller behaves like a sort of input/output timed automata [3] interpreter. Reading a timed input event updates the state of the running automaton (the automaton is reactive). After some delay, depending on the active state, a default transition is always activated by sending back a timed planned event to the system (the automaton is time active).

What exact type of timed automata are to be executed by the music controller layer ? This is still a matter of research. Our proposal provides some indication of *how* to run an interactive score. The true nature of an interactive score is still an open question.

6. CONCLUSION

For an effective use of this proposed system we now need to gain a deeper understanding of how the music controller can be programmed. At the operational level, music controllers look like timed automata. But this fairly low level model seems inadequate for interactive music composition.

There is still a need to develop a high level modular language for the description of interactive scores. This might be achieved by pursuing the research which led to proposals such as *iScore* [2]. In particular, we may consider the following three complementary research projects.

The first concerns the capacity of such a score to describe musical anticipation in a simple fashion as one of the main conceptual tools used in music composition.

Already in the 80's some proposals emerged in this direction [6]. But there are still many questions to be answered. Aside statistical analysis and continuation techniques that are proposed by softwares such Continuator [16] or OMax [9], we also believe that structural analysis of musical languages may be conducted. For instance, musical anticipation may be envisaged, on a more abstract level than music scores, as a generalization of musical anacrusis [12].

The second concerns the various combinations possible of interactive programs which may be defined. The sequential and parallel composition of elements of interactive scores are obviously required. But what type of sequential composition ? What type of parallel composition ? How may input events be distributed among the different elements of these scores ? Are they to be duplicated ? Buffered ? An initial study of sequential composition, both from the perspective of music modeling [12] or from a purely theoretical point view [13, 14] already shows that, together with anticipation modeling, a lot remains to be said.

The third concerns the hierarchical description of the music. It seems that composers are looking for ways of thinking about their music on several levels of abstraction : say from elementary sounds to performance in a concert via musical motifs, movements, pieces, etc...

Hierarchical system modeling techniques have already been defined in various areas in computer science. In particular, statecharts in UML [10] is based on a hierarchical description of this type. However, standard statecharts semantics may need to be adapted for hierarchical interactive music descriptions.

These ideas for future research, on the musical side of our proposal, also need to be combined with existing techniques and concepts for low level audio stream analysis (as inputs) or audio stream production (as outputs) or with those still to be developed.

Synchronizing two elements from scores that result from real-time computation issuing from the history of the global piece being performed is one thing. Combining the associated audio streams these elements realize is quite

another. It seems that each operator defined on the symbolic side of music demands a counterpart on the realization side.

In all cases, we expect that the present proposed system architecture will facilitate further experimentation.

7. REFERENCES

[1] Bresson J. Agon C. and Assayag G. *The OM composer's Book, Vol.1 & Vol.2*. Collection Musique/Sciences. Ircam/Delatour, 2006.

[2] Antoine Allombert, Myriam Desainte-Catherine, and Gérard Assayag. Iscore : a system for writing interaction. In *Third International Conference on Digital Interactive Media in Entertainment and Arts (DIMEA 2008)*, pages 360–367. ACM, 2008.

[3] Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2) :183–235, 1994.

[4] Alessandro Cipriani and Maurizio Giri. *Electronic Music and Sound Design - Theory and Practice with Max/Msp*. Contemponet, 2010.

[5] Arshia Cont. Antescofo : Anticipatory synchronization and control of interactive parameters in computer music. In *International Computer Music Conference (ICMC)*, 2008.

[6] P. Desain and H. Honing. Loco : a composition microworld in logo. *Computer Music Journal*, 12(3) :30–42, 1988.

[7] D. Fober, Y. Orlarey, and S. Letz. Faust architectures design and OSC support. In *14th Int. Conference on Digital Audio Effects (DAFx-11)*, pages 231–216. IRCAM, 2011.

[8] Alexandre R. J. François and Elaine Chew. An architectural framework for interactive music systems. In *International Conference on New Interfaces for Musical Expression*, pages 150–155, 2006.

[9] M. Chemillier G. Assayag, G. Bloch. Omax-ofon. In *Sound and Music Computing (SMC) 2006*, 2006.

[10] David Harel. Statecharts in the making : a personal account. In *Proceedings of the Third ACM SIGPLAN History of Programming Languages Conference (HOPL-III), San Diego, California, USA, 9-10 June 2007*, pages 1–43. ACM, 2007.

[11] Thomas A. Henzinger. The theory of hybrid automata. In *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 278–292. IEEE Computer Society, 1996.

[12] David Janin. Modélisation compositionnelle des structures rythmiques : une exploration didactique. Technical Report RR-1455-11, LaBRI, Université de Bordeaux, August 2011.

[13] David Janin. On languages of one-dimensional overlapping tiles. Technical Report RR-1457-12, LaBRI, Université de Bordeaux, January 2012.

[14] David Janin. Quasi-recognizable vs MSO definable languages of one-dimentionnal overlaping tiles. Technical Report RR-1458-12, LaBRI, Université de Bordeaux, February 2012.

[15] S. Letz, Y. Orlarey, and D. Fober. Real-time composition in Elody. In *Proceedings of the International Computer Music Conference*, pages 336–339. ICMA, 2000.

[16] F. Pachet. The continuator : Musical interaction with style. In *Proceedings of ICMC*, pages 211–218, Göteborg, Sweden, September 2002. ICMA. best paper award.

RÉTROACTION MUSIQUE - PHYSIOLOGIE : UNE APPROCHE SELON LE PARADIGME DES ÉMOTIONS

Pierre-Henri Vulliard, Joseph Larralde, Myriam Desainte-Catherine
Univ. Bordeaux, LaBRI, UMR 5800, Talence, France
CNRS, LaBRI, UMR 5800, Talence, France
{phvulliard, joseph.larralde}@gmail.com
myriam.desainte-catherine@labri.fr

RÉSUMÉ

Il est un fait avéré que l'écoute de musique induit des réactions physiologiques particulières chez l'auditeur, et l'étude de ces inductions constitue un terrain d'études encore vaste. Lorsque l'on veut analyser les signaux physiologiques mesurés sur une personne écoutant de la musique, il faut définir des modèles pour savoir quelles informations rechercher dans ces signaux. Inversement, lorsque l'on cherche à générer de la musique à partir de signaux physiologiques échantillonnés, on cherche en fait à créer une induction inverse de celle qui a lieu naturellement, et il faut pour cela définir des modèles afin d'être capable de contrôler tous les paramètres d'un système de musique générative à partir des quelques signaux physiologiques à disposition, et ce de manière cohérente. La notion d'émotion, en plus de sembler toute indiquée dans le contexte, se révèle être une notion pivot très pratique pour faire correspondre des modèles musicaux avec des modèles physiologiques.
Nous proposons dans cet article un système temps-réel expérimental visant à étudier les interactions et rétroactions entre musique et physiologie, basé sur le paradigme des émotions.

1. INTRODUCTION

Le système expérimental présenté dans cet article, baptisé MuZICO, permet d'aborder les interactions entre musique et physiologie selon deux approches : l'une, de bas niveau, se base sur la sonification des signaux physiologiques issus du système nerveux autonome, créant ainsi un lien direct entre l'expression inconsciente d'un état physiologique et la production de vibrations acoustiques. L'autre approche, de haut niveau, introduit une couche d'abstraction prenant en charge la reconnaissance d'états de vigilance (vigilance, relaxation, sommeil) et/ou émotionnels de l'auditeur, et permet de piloter des paramètres de génération musicale de haut niveau en fonction de ces états.
MuZICO utilise actuellement pour la capture de signaux physiologiques un système de biofeedback Thought Technologies ProComp5 infinity. Il s'agit d'un boîtier à 5 entrées sur lesquelles on peut connecter différents types de capteurs.
Les capteurs utilisés avec MuZICO sont les suivants :

- capteur de respiration abdominal / thoracique
- capteur de conductivité de la peau
- électrocardiographe
- électroencéphalographe

Par la suite, nous décrivons l'évolution de nos travaux sur la sonification et l'interaction entre les états de vigilance et la synthèse musicale, puis nous proposons une généralisation des états de vigilance aux états émotionnels et sa validation dans le cadre d'une collaboration avec une équipe effectuant des recherches sur la reconnaissance d'émotions exprimées corporellement. Enfin, nous étudions les perspectives offertes par la synthèse de ces expériences pour l'étude des interactions et rétroactions entre musique, émotions perçues, et physiologie.
Ces travaux se sont déroulés dans le cadre du projet SCRIME, conventionné par l'Université Bordeaux 1, l'Institut Polytechnique de Bordeaux et le Conservatoire de Musique de Bordeaux, et financé par la DGCA du Ministère de la Culture.

2. TRAITEMENT DE SIGNAUX PHYSIOLOGIQUES SELON L'APPROCHE ÉNERGÉTIQUE

2.1. Sonification : mise en correspondance directe

La sonification joue dans notre cas un rôle de catalyseur de l'attention de l'auditeur et de sa compréhension du lien entre les manifestations de son système nerveux autonome, habituellement involontaires, et la musique qu'elles génèrent. L'énergie déployée par les organes internes et échantillonnée grâce aux capteurs est traduite directement en énergie sonore, créant ainsi un phénomène de perception synesthésique.
Concrètement, nous calculons les vitesses de variation des données provenant de tous les capteurs, excepté des électrodes EEG. Ces vitesses de variation sont traduites en énergie par le contrôle direct du volume, du timbre ou de la fréquence d'un son. En ce qui concerne les capteurs cardiaques et de respiration, la fréquence du cycle est évaluée (fréquence des battements de cœur ou des cycles inspiration / expiration), et les variations de cette fréquence sont également exprimées par des variations de volume,

de timbre ou de hauteur. En ce qui concerne les signaux EEG, l'évaluation de l'évolution des rapports énergétiques entre des plages de fréquence caractéristiques de certains états de vigilance [6] est sonifiée de la même manière.

2.2. Interaction vigilance / synthèse musicale

Outre la production directe du son à partir des signaux physiologiques (approche orientée instrument), l'approche énergétique passe également par un plus haut niveau d'abstraction en analysant l'état d'éveil de l'auditeur, pour générer un flux musical à partir de règles de composition harmoniques, rythmiques et timbrales. L'idée est ici de contrôler le dynamisme de ce flux musical en fonction de l'état d'éveil de l'auditeur.
Les données physiologiques sont donc intégrées à un seul paramètre continu représenté par un axe et représentant l'état d'éveil de l'auditeur, lui-même étant ensuite redistribué sur l'ensemble des paramètres de génération musicale : il s'agit d'un mapping $M \rightarrow 1 \rightarrow N$.
Plusieurs travaux menés par Pierrick Legrand et Frédérique Faïta Ainseba [8] montrent le lien entre certains paramètres de composition musicale, l'énergie évoquée par la musique résultante, et le niveau de vigilance induit par son écoute chez un auditeur.
Ces paramètres, dans leur mise en œuvre au sein de l'environnement MuZICO, ont suivant les cas des valeurs discrètes ou continues, définies par des adjectifs ou des valeurs numériques. Par exemple le timbre d'un son peut aller de étouffé à brillant, ce qui correspond à une grandeur énergétique mesurable dans son signal, liée à son centroïde spectral. Les autres paramètres liés à l'énergie sont le tempo de la musique, l'attaque de l'enveloppe dynamique des notes, la hauteur des notes, le nombre d'instruments simultanés, le côté plus ou moins "naturel" des sons instrumentaux, la nature percussive ou entretenue des sons, ainsi que le nombre de notes simultanées [5].
Tous ces paramètres sont caractérisés par des intervalles bornés utilisés par MuZICO pour la génération musicale.

Soient V_Emin et V_Emax les intervalles de valeurs de paramètres musicaux correspondant respectivement à des énergies évoquées par la musique faibles et élevées :

Paramètre (unité)	V_Emin	V_Emax
tempo (bpm)	[0.1 ; 60]	[144 ; 300]
timbre	étouffé	brillant
attaque des notes (ms)	[100 ; 1000]	[1 ; 30]
hauteur des notes (Hz)	[110 ; 220]	[440 ; 880]
sons instrumentaux	naturels	synthétiques
nature des sons	percussive	entretenus
nombre de voix	1	beaucoup
densité de notes	faible	élevée

Les paramètres ayant une unité indéfinie se traduisent au sein de MuZICO dans différentes unités en fonction de l'algorithme de production sonore permettant leur implémentation (synthèse, lecture d'échantillons, contrôle d'effet). Par exemple, on pourra traduire la brillance d'un son par un réglage de l'index de modulation dans un algorithme de synthèse FM, ou par le contrôle de la fréquence de coupure d'un filtre passe-bas sur un lecteur d'échantillons.

Figure 1. Pierre-Henri Vulliard au Marché de Lerme.

2.3. Applications

Le système MuZICO a déjà été présenté au public dans des cadres artistiques (utilisation en concert comme accompagnement musical génératif) et de vulgarisation scientifique (présentation sous forme d'atelier ou de borne interactive). Ses applications peuvent donc être artistiques, pédagogiques, ou thérapeutiques.

Dans le cadre artistique, MuZICO a été utilisé dans deux configurations : l'une, impliquant fortement l'un des auteurs en tant que musicien saxophoniste, consiste à générer un accompagnement à partir de capteurs de respiration et cardiaques ou eeg en tant que support d'improvisation pour le musicien. Concrètement, cette configuration a été utilisée lors de concerts de musique électroacoustique organisés par le SCRIME (au Marché de Lerme de Bordeaux (voir figure 1), et à l'Hôtel de Région d'Aquitaine), ou de performances (au musée scientifique Cap Sciences et au club i-Boat à Bordeaux). L'autre a été mise en place dans le cadre du projet ANR Care et permet à un danseur (Gaël Domenger) muni d'une combinaison de capture de mouvements de piloter lui-même la génération de la musique sur laquelle il improvise (voir figure 2). Une représentation publique a eu lieu au Casino de Biarritz en mars

Figure 2. Gaël Domenger utilisant eMotion et MuZICO.

2011.

Dans un cadre plus pédagogique, des séances de décryptage du système MuZICO ont été présentées au public de manière ponctuelle : des ateliers ont eu lieu en durant les journées Eurêka de la Fête de la Science 2011 à l'Hôtel de Région d'Aquitaine et pendant la Semaine du Son 2012 à Cap Sciences. D'une manière plus suivie, une borne interactive permettant d'essayer le système a été mise en place à Cap Sciences pendant toute la durée de l'exposition Numériquement Vôtre (mars - décembre 2011). Le public pouvait interagir avec MuZICO grâce à un capteur de conductivité de la peau intégré à la borne.

Dans le domaine thérapeutique, des investigations sur l'apport possible du système aux techniques de relaxation sont menées parallèlement depuis le début du projet par un des auteurs auprès de professionnels du milieu tels que des masseurs ou des hypnotiseurs.

3. INTERACTION ÉMOTIONS / SYNTHÈSE MUSICALE

3.1. La représentation des émotions

Nous proposons ici une généralisation des états de vigilance aux états émotionnels dans le but d'augmenter les possibilités d'expressivité de la génération musicale, en ajoutant à l'axe de la vigilance un deuxième axe. Il y a beaucoup de définitions et de représentations des émotions dans le champ d'étude de la psychologie, mais peu d'entre elles sont actuellement utilisées en informatique.
Nous avons choisi d'utiliser le même modèle que dans [5] et [9], modèle couramment employé en psychologie sous le nom de roue des émotions de Plutchik [7]. Il s'agit d'un espace bi-dimensionnel défini par deux axes correspondant à l'excitation (que nous assimilons à la vigilance dans le cadre de la génération musicale), et à la valence (plaisir - déplaisir), et dans lequel nous pouvons placer les émotions primaires d'Ekman [3] (voir figure 3).

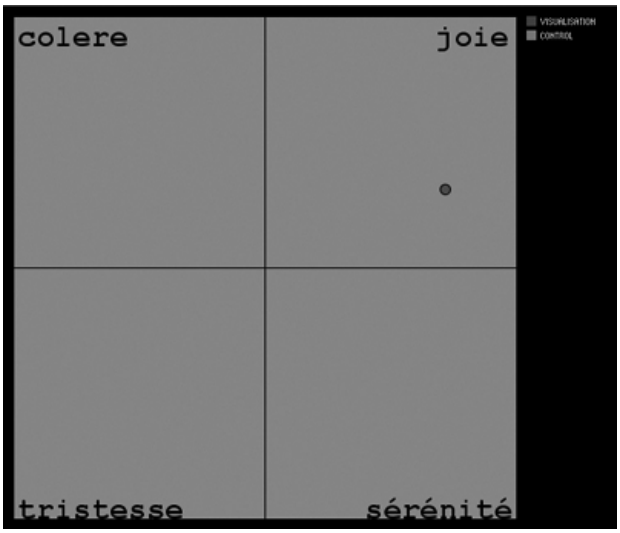


Figure 3. placement des émotions primaires d'Ekman dans l'espace 2D de représentation des émotions.

3.2. Projet Care - logiciel eMotion

La collaboration dans le cadre du projet Care [2] avec une équipe effectuant des recherches sur la reconnaissance d'émotions basées sur le mouvement, nous a permis de tester la génération de la musique en utilisant les deux axes de l'espace des émotions.
Dans le cadre du spectacle de danse servant de conclusion à ce projet, nous avons généré un flux sonore et musical en utilisant l'approche énergétique décrite précédemment pour associer les règles de composition musicale aux étiquettes d'émotions envoyées par le logiciel de reconnaissance d'émotions eMotion. Ce logiciel effectuant ses analyses à partir des informations de position et de mouvement envoyées par une combinaison munie de capteurs, nous avons pu aussi utiliser l'approche instrumentale pour associer cette fois-ci des sons directement aux mouvements du danseur.
L'analyse de questionnaires distribués au public lors de la représentation au Casino de Biarritz a ensuite permis une réflexion a posteriori sur l'expérience.

3.3. Validation du système par le public

La deuxième représentation du spectacle de restitution du projet Care s'est faite devant un public de 120 personnes. A leur arrivée, ces personnes se sont vues remettre chacune un questionnaire dans le but pour l'équipe de collecter les retours des spectateurs [1]. 97 questionnaires ont été remplis et rendus, soit approximativement 80% de l'ensemble des questionnaires distribués. Pour un non-chercheur, le monde scientifique peut parfois paraître quelque peu hermétique. Le spectacle présentait les résultats du projet Care sous une forme à la fois artistique et ludique, les scientifiques étaient sur scène, ce que le public a apprécié. 90% des personnes qui se sont exprimées sur le questionnaire ont cité l'aspect innovant et magique du spectacle comme le meilleur point. Cependant, 75% de ces

mêmes personnes ont trouvé que la musique générée évoquait de manière trop évidente l'émotion exprimée par le danseur.
Cette critique est intéressante car elle valide les paramètres choisis pour la génération musicale.

3.4. Valence et complexité musicale

D'après [5], les paramètres musicaux liés à la valence des émotions évoquées par la musique sont la tonalité et la complexité. Nous proposons ici des modèles génératifs harmoniques, rythmiques et mélodiques, ainsi qu'une description de leurs paramètres sous l'angle de vue de la complexité musicale.
La manière dont ces modèles (implémentés sous forme de modules dans MuZICO) communiquent entre eux est explicitée en appendice.

3.4.1. Échelles de hauteurs

Nous proposons un modèle de génération d'échelles de hauteurs basé sur la suite :

$$U_{n+1} = \left(U_n \times \omega^{\frac{V_n \bmod (m+1)}{\theta}} - U_0 \right) \times \omega^{-p}$$
$$où\ U_0 = f_f, et\ p \in \mathbb{N}^+ tel\ que\ U_0 < U_{n+1} < \omega \times U_0 \quad (1)$$

où ω est le rapport d'octave, θ le nombre par lequel on divise l'octave pour obtenir le tempérament, f_f la fréquence fondamentale de la gamme générée, V un vecteur de dimension m, et $V_{i,i\in[1..m]}$ la $i^{ème}$ composante de V.

On considère que la complexité d'une échelle de hauteurs dépend du nombre d'itérations nécessaires à sa génération, et des valeurs de ω, θ, et V.
Par exemple, on se place dans le cas où $\omega = 2$, $\theta = 12$ (musique tonale occidentale).
Soit $V = (7)$ (génération de l'échelle de hauteurs par parcours du cycle des quintes), et N le nombre d'itérations effectuées. On a alors :

- la gamme pentatonique ou l'accord Maj 6/9 pour N = 4
- la gamme diatonique pour N = 6
- la gamme chromatique pour N = 11

Ces gammes ont une complexité harmonique croissante. Des échelles de complexités différentes peuvent aussi être créées en faisant varier V : soit $V = (4)$ et $N = 2$, on génère alors une échelle de hauteur correspondant à un accord augmenté. Soit $V = (3)$ et $N = 3$, on génère alors une échelle de hauteurs correspondant à un accord diminué. Soit $V = (3, 4, 4, 3)$ et $N = 6$, on génère ainsi la gamme mineure mélodique ascendante.

3.4.2. Suites d'accords

Dans MuZICO, les suites d'accords sont générées par une grammaire qui tient compte du contexte pour supervise les transpositions liées aux modulations. La complexité d'un accord dépend de plusieurs facteurs que nous avons identifié :

- le nombre de notes qui le constituent
- la complexité de l'échelle de hauteurs sur laquelle il est construit
- l'ordre d'apparition de ses notes dans la construction de cette échelle par itérations

La complexité d'une suite d'accords dépend également de plusieurs facteurs :

- la complexité des accords qui la constituent
- le nombre d'accords différents dans la séquence
- la complexité harmonique des enchaînements d'accords, en prenant en compte principalement les cadences et les résolutions dans le cadre de la musique modale et tonale, et les modulations dans le cadre de cette dernière.

3.4.3. Motifs rythmiques

Les motifs rythmiques sont créés par superposition de plusieurs patterns rythmiques linéaires synchronisées à une même pulsation et de longueur définie, chaque pattern rythmique étant défini par un décalage par rapport à l'instant initial commun, et par une densité rythmique (nombre de pulsations entre deux coups du pattern). Cette superposition se fait par une opération de "ou" logique, si l'on considère les patterns rythmiques superposées comme des vecteurs de bits, 1 représentant un coup et 0 le silence.
Soient m la longueur des patterns rythmiques, q le nombre de patterns superposés, of_i le décalage par rapport à l'instant initial du $i^{ème}$ pattern, et d_i la densité rythmique du $i^{ème}$ pattern.
Si l'on considère qu'un vecteur de bits est équivalent à un ensemble fini de nombres entiers, chaque nombre de l'ensemble indiquant la présence d'un bit positif à l'indice lui correspondant dans le vecteur, un motif rythmique R constitué par superposition de patterns P_i peut alors être défini par :

$$R = \bigcup_{i=0}^{q} P_i$$
$$où\ P_i = \bigcup_{n=0}^{E((m-of)/d)} (of_i + d_i \times m) \quad (2)$$

En ce qui concerne la complexité rythmique, nous utilisons un modèle basé sur la mesure de complexité de Toussaint ou autres techniques d'évaluation de complexité rythmique, qui attribue un poids à chaque pulsation (voir figure 4).

3.4.4. Génération de mélodies

Pour générer la mélodie, MuZICO est doté d'un module prenant en entrée l'échelle de hauteurs courante et plusieurs motifs rythmiques générés selon les techniques expliquées précédemment. Chaque note a un poids correspondant à son ordre d'apparition dans la construction de l'échelle de hauteurs courante. Pour chaque coup du motif rythmique, une note est tirée au hasard selon une densité

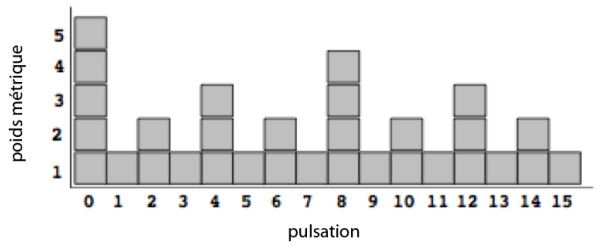


Figure 4. poids des temps dans un pattern rythmique de taille 16.

de probabilité gaussienne variant en fonction du poids métrique correspondant. On obtient ainsi un profil mélodique initial qui pourra ensuite être développé musicalement par divers algorithmes tels que le PST (Probabilistic Suffix Tree). Ce profil est ensuite modifié en fonction des informations de modulation transmises par la grammaire générant les accords pour donner la mélodie qui sera jouée.
La complexité d'une mélodie jouée de cette manière dépend des densités de probabilités gaussiennes et des paramètres de configuration du PST.

4. CONCLUSION ET DISCUSSION

Le logiciel MuZICO a montré sa pertinence, d'une part dans la génération de musique en fonction des mesures physiologiques de la vigilance, d'autre part dans la génération de musique en fonction des émotions exprimées et représentées dans un espace à deux dimensions (énergie/valence). Une prochaine étape consiste donc à trouver à partir des données électroencéphalographiques un nouveau critère, soit par la mesure de l'évolution de la fréquence ou de l'amplitude des ondes cérébrales en un site correspondant à une activité cognitive, émotionnelle ou perceptive particulière (par exemple perception de la consonnance), soit par la mesure de l'évolution comparée des fréquences ou des amplitudes des ondes cérébrales de points situés sur chacun des hémisphères cérébraux. Nous entamons actuellement dans cette optique une collaboration avec des chercheurs en Interface Cerveau-Ordinateur. Nous pourrons ainsi générer la musique en fonction des émotions réellement ressenties et non plus exprimées par l'auditeur, et donc optimiser de manière significative la rétroaction, pour améliorer à la fois le pouvoir d'évocation de la musique et ses qualités esthétiques.
Une autre étape à venir sera d'affiner notre système au niveau de la synthèse sonore en faisant correspondre de manière plus précise aux émotions les paramètres musicaux de plus bas niveau (correspondant aux paramètres perceptifs des sons utilisés ou générés), par une meilleure gestion des bornes, en se conformant à un namespace au format url similaire à celui utilisé dans le protocole OSC.
Une piste de validation alternative pour le projet est d'utiliser des algorithmes d'apprentissage par renforcement [4] afin de déterminer les combinaisons de paramètres les plus évocatrices d'émotions particulières. Des expérimentations de cet ordre sont en cours depuis le début du projet avec ce type d'outils logiciels.

Remerciements

Nous remercions Pierre Héricourt, ingénieur au service système du LaBRI, pour le développement du driver permettant la liaison entre le boîtier ProComp5 et les différents logiciels utilisés (Pure Data pour la génération du son et l'analyse des données physiologiques, et OpenVibes, bibliothèque d'outils logiciels d'interaction cerveau-ordinateur, pour une analyse plus fine des signaux EEG).

5. REFERENCES

[1] A. Clay, N. Couture, E. Decarsin, M. Desainte-Catherine, P-H. Vulliard, and J. Larralde. Movement to emotions to music : using whole body emotional expression as an interaction for electronic music generation. 2012.

[2] A. Clay, G. Domenger, N. Couture, J.B. De La Rivière, M. Desainte-Catherine, et al. Spectacle augmenté : le projet care, un processus de recherche. 2011.

[3] P. Ekman. An argument for basic emotions. *Cognition & Emotion*, 6(3-4) :169–200, 1992.

[4] S. Le Groux and P.F.M.J. Verschure. towards adaptive music generation by reinforcement learning of musical tension. 2010.

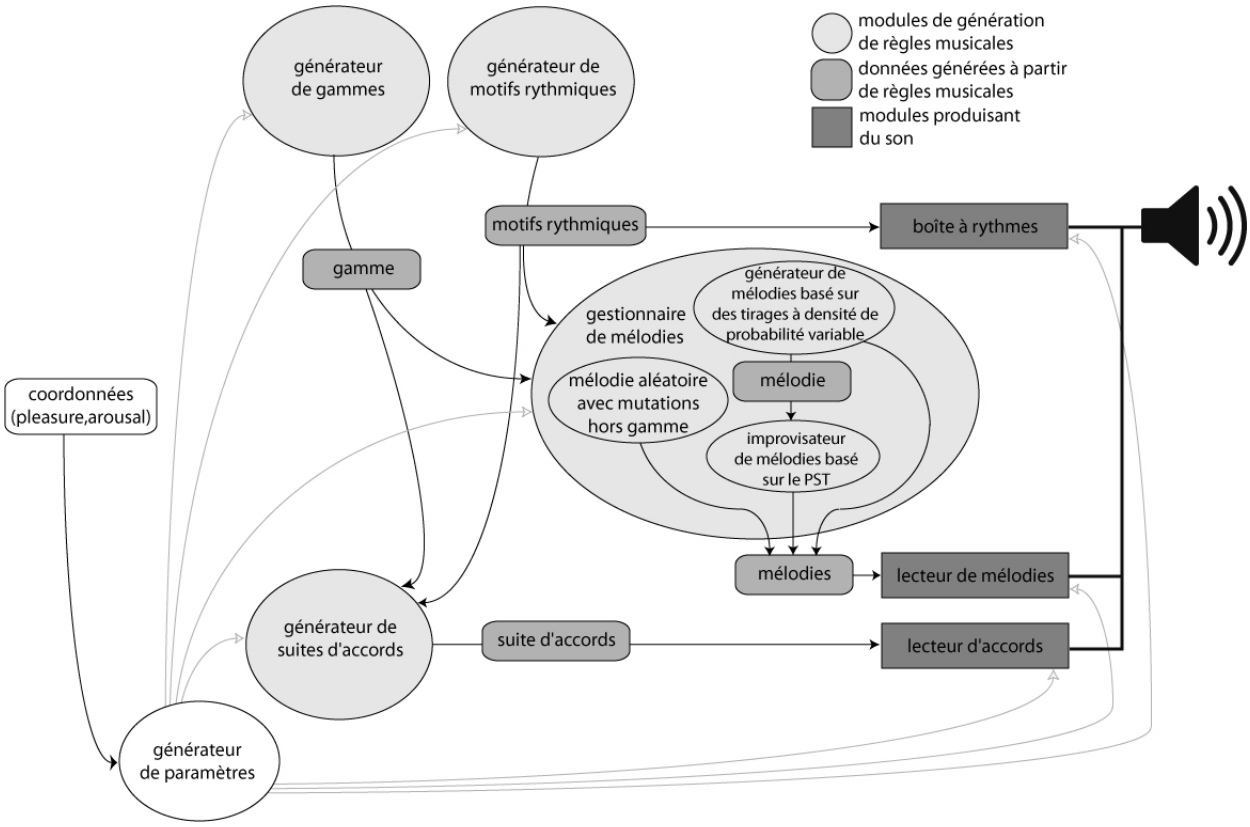
[5] S.R. Livingstone, R. Muhlberger, A.R. Brown, and W.F. Thompson. Changing musical emotion : A computational rule system for modifying score and performance. *Computer Music Journal*, 34(1) :41–64, 2010.

[6] J. Paty. Les états de conscience. *Psychologie médicale*, 1985.

[7] J.A. Russel. A circumplex model of affect. *Journal of Personnality and Social Psychology*, 39(6) :1161–1178, 1980.

[8] L. Vezard, M. Chavent, P. Legrand, F. Faita-Ainseba, J. Clauzel, et al. Caractérisation d'états psychophysiologiques par classification de signaux eeg. intégration de ces résultats dans le projet psi. 2011.

[9] I. Wallis, T. Ingalls, and E. Campana. Computer-generating emotional music : The design of an affective music algorithm. In *International Conference on Digital Audio Effects*, pages 7–12, 2008.



Annexe. Schéma fonctionnel de la partie de MuZICO gérant les règles de composition musicale.

IMPROTEK : INTÉGRER DES CONTRÔLES HARMONIQUES POUR L'IMPROVISATION MUSICALE DANS LA FILIATION D'OMAX

Jérôme Nika
Ircam - Paris, puis Télécom ParisTech
46 rue Barrault - 75013 Paris
jerome.nika@telecom-paristech.fr

Marc Chemillier
Centre d'analyse et de mathématique sociales
EHESS, 190 avenue de France - 75013 Paris
chemilli@ehess.fr

RÉSUMÉ

Le système ImproteK présenté dans cet article propose d'intégrer un cadre rythmique et une structure harmonique sous-jacente dans un contexte d'improvisation. Dans la filiation du logiciel d'improvisation OMax [4, 3, 13], il repose sur la structure d'oracle des facteurs pour tirer profit des propriétés particulièrement riches et pertinentes de cet automate dans un contexte musical [5]. Au cours d'une session d'improvisation, ce système peut s'adapter à une pulsation régulière et proposer des phrases musicales suivant une progression harmonique donnée. ImproteK est conçu comme un instrument interactif dédié à la performance : ses improvisations sont construites à partir de la modélisation stylistique réalisée sur le jeu *live* d'un musicien ou sur un corpus *offline*. Associée à des techniques considérant le corpus comme une mémoire musicale, cette modélisation s'étend aux domaines de l'harmonisation et de l'arrangement dans un module d'interaction harmonique.

1. INTRODUCTION

Le propos d'ImproteK est d'allier modélisation stylistique et interaction pour mettre en place un dialogue original entre des musiciens et un improvisateur virtuel nourrissant sa propre inspiration de leur jeu. Ces deux paradigmes et la structure d'automate au coeur de son implémentation (section 2) font de ce système un cousin du logiciel d'improvisation OMax [4, 3, 13] conçu et développé à l'Ircam.

Dans la lignée des travaux sur la modélisation du style menée par G. Assayag et. al [6], OMax apprend le style d'un improvisateur humain grâce à une représentation basée sur la structure d'oracle introduite par C. Allauzen et. al [1] et étendue à un contexte musical [5]. Le logiciel élabore un modèle du jeu du musicien en temps réel, et est ensuite à même de naviguer à travers cette représentation en suivant des chemins différents de ceux empruntés par son partenaire pour créer ainsi de nouvelles improvisations partageant la même esthétique.

Partageant cette intention, ImproteK se concentre sur un contexte d'improvisation musicale associée à une structure harmonique sous-jacente et présentant une pulsation régulière. Le système propose une interaction enrichie en prenant la pulsation en considération dans le cadre d'une grille d'accords donnée (section 3). Cette conception de l'improvisation est concrètement réalisée par le moyen d'une architecture permettant son intégration au sein d'une formation musicale dont le tempo peut être extrait et suivi (section 4). Enfin, ImproteK offre une interaction harmonique en étendant la modélisation stylistique aux domaines de l'harmonisation et de l'arrangement (section 5).

Cet article rappellera premièrement les principes généraux partagés par OMax et ImproteK pour décrire ensuite les développements spécifiques à ce dernier.

2. MODÉLISATION DU STYLE MUSICAL ET STRUCTURE D'ORACLE

2.1. Modélisation stylistique pour l'improvisation et l'harmonisation

Depuis M&Jam Factory [21], Cypher de R. Rowe [18], ou Voyager de G. Lewis [14], souvent considérés comme les premiers systèmes interactifs en temps réel, de nombreux "partenaires virtuels pour l'improvisation" ont été développés. La plupart d'entre eux ont bénéficié du développement des techniques d'apprentissage automatique avec l'idée grandissante d'arriver toujours plus près du discours musical tenu par l'improvisateur humain en interaction.

Parmi ces dispositifs, le Continuator [16] conçu par F. Pachet modélise l'entrée musicale à l'aide d'un modèle Markovien étendu pour créer de nouvelles phrases à partir de cet apprentissage. Comme dans le mode *free* d'OMax, on ne trouve aucun mécanisme de perception rythmique permettant une synchronisation avec le musicien. A l'inverse, GenJam [7] développé par J. Biles est plus proche des expériences antérieures d'OMax avec un mode *beat* : le logiciel propose un accompagnement comportant un tempo fixé pour guider l'improvisation du musicien humain lors de phases d'écoute. Il répète ensuite certaines des séquences entendues après avoir apporté des modifications à l'aide d'un algorithme génétique. Band Out of a Box de B. Thom [20] implique également un accompagnateur non-interactif avec un tempo imposé dans un schéma d'interaction basé sur une logique de "trading fours" : un improvisateur humain et son partenaire virtuel dialoguent en question-réponse sur des séquences de 4 mesures. Chacune des mesures jouées par l'improvisateur

humain est attribuée à une classe appelée *playing mode*, et la réponse de la machine est composée de 4 mesures appartenant à la même suite de modes.

Ce dernier exemple introduit le sujet récurrent de la meilleure segmentation de l'entrée musicale et de sa représentation. Cet élément est particulièrement crucial dans le cas de systèmes s'attelant à l'harmonisation, l'arrangement ou plus généralement l'accompagnement en faisant appel à un corpus. Qu'ils soient interactifs et dédiés à la performance ou conçus pour être utilisés "offline", ils peuvent être classés schématiquement selon deux critères : l'utilisation exclusive de règles musicales formalisées et l'implication d'un corpus musical (voir [15] pour la situation d'ImproteK au sein d'un système de classification plus exhaustif). Dans le cas de cette dernière catégorie, le corpus peut être considéré comme un environnement d'apprentissage pour créer des modèles génératifs (par exemple C. Chuan et E. Chew [11], le logiciel Mysong-Songsmith de Microsoft [19]) et/ou comme une mémoire musicale dans laquelle des fragments sont recherchés et réagencés pour composer le nouveau matériau (bassiste de jazz virtuel de G. Ramalho, ImPact [17]).

La définition de l'unité de segmentation pour le traitement du corpus est un élément caractéristique de l'approche de ces différents systèmes : le *grain* peut en effet correspondre à quelques notes clés d'une mélodie réduite comme chez C. Chuan et E. Chew, à un seul accord pour les logiciels Songsmith et Band in a Box (PG Music), ou à certaines cadences ou séquences d'accords dans le projet ImPact de G. Ramalho. L'idée est de trouver le meilleur équilibre entre des tranches assez longues pour proposer un accompagnement plausible et cohérent, et des tranches assez fines pour éviter de recopier des fragments trop longs et donc trop identifiables dans le corpus.

Dans le cas d'Improtek, l'unité choisie est la *pulsation*. En effet, le modèle d'automate structurant sa mémoire - l'oracle des facteurs - permet de s'affranchir du choix entre cohérence musicale et originalité par rapport au corpus grâce à la continuité qu'il assure par construction, permettant ainsi de travailler avec un grain aussi fin.

2.2. Oracle et reconnaissance de motifs

L'utilisation première de l'oracle des facteurs a été la reconnaissance de motifs dans des chaînes de caractères, pour être ensuite étendue au calcul de facteurs répétés dans un mot et à la compression de données. Cet automate acyclique représente au moins tous les facteurs d'un mot, et l'algorithme de construction incrémentale est linéaire en sa longueur, en temps comme en espace (pour le détail de l'algorithme, se reporter à [1]).

Les propriétés formelles de la structure d'oracle sont détaillées dans les articles fondateurs, et pleinement appliquées à la problématique de *réinjection stylistique* [4] dans plusieurs documents consacrés à OMax. Aussi, seuls les outils assurant la continuité et la cohérence des séquences générées avec un oracle sont brièvement décrits ici : les transitions et les liens suffixiels.

Les *transitions* (liens en avant, lignes pleines) permettent d'atteindre chaque sous-séquence (ou facteur) de la séquence d'origine en partant de l'état initial. De cette manière, chaque progression entre deux états consécutifs de la suite d'origine peut être générée.

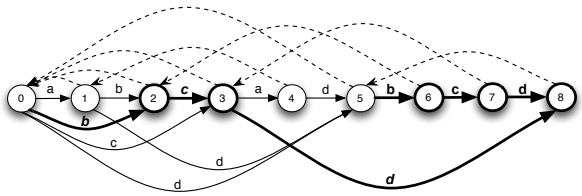


Figure 1. Oracle pour la séquence *abcadbcd* : générer un facteur depuis l'état initial.

La figure 1 présente l'oracle construit sur la séquence *abcadbcd* et illustre ce point en affichant un chemin utilisant des transitions pour produire la sous-séquence *bcd* (0, 2, 3, 8), présente à l'origine entre les états 5 et 8.

L'oracle repère également les motifs répétés dans la séquence d'origine à l'aide des *liens suffixiels* (liens en arrière, lignes pointillées). Ils pointent sur l'état final des occurrences précédemment rencontrées d'un motif.

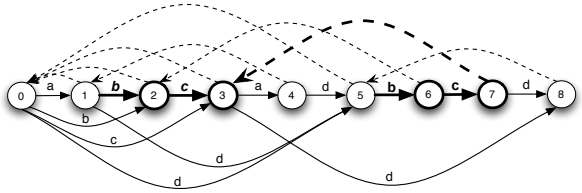


Figure 2. Oracle pour la séquence *abcadbcd* : exemple du motif répété *bc*.

Dans l'exemple de la figure 2, un lien suffixiel connecte les états 7 et 3. 3 est en effet l'état le plus à gauche où le plus long suffixe répété de la séquence terminant en 7 (*abcadbc*) est reconnu : *bc*.

La structure d'oracle peut ainsi être vue comme un outil pour créer de nouvelles séquences grâce à une navigation non-linéaire empruntant ces deux types de liens, sa construction lui permettant de dégager la logique dans les progressions présentes dans les chaînes. Dans le cas d'une application musicale, son aptitude à préserver le discours du matériau original permet de développer une esthétique simultanément nouvelle et proche de celle proposée par les musiciens en interaction.

3. IMPROVISER DANS UN CONTEXTE MÉTRIQUE ET HARMONIQUE

3.1. Oracles en *Mode beat*

Le principe de la version actuelle du logiciel OMax est de faire de chaque élément musical isolé un état dans une instance de la structure d'oracle. La navigation *libre* [2]

en temps réel à travers une mémoire ainsi constituée en fait donc un instrument interactif dédié à l'improvisation dans un contexte de musique *libre*. L'approche d'ImproteK diffère par l'intégration de deux paradigmes. Premièrement, ses improvisations prennent place dans le cadre d'une structure harmonique représentée par une progression symbolique choisie au début de chaque session (*grille d'accords*), lui permettant ainsi d'étendre la modélisation stylistique aux domaines de l'harmonisation et de d'arrangement. Ensuite, il intègre un cadre métrique en instituant la pulsation comme unité élémentaire de ses acquisitions, restitutions, et générations. Pour ce faire, le module d'improvisation a été développé sur la base d'une version précédente d'OMax (OMax 2.0, 2004) conçue comme une bibliothèque Lisp sous l'environnement OpenMusic [9]. Cette version implémentait la structure d'oracle aussi bien pour un contexte d'improvisation libre (mode *free*, dimension retenue dans la version actuelle d'OMax), que pour le cas d'une pulsation régulière (mode *beat*, point de départ du développement d'ImproteK).

Dans cette seconde optique, chaque état d'un oracle représente une tranche musicale dont la durée est donnée par le tempo du morceau en cours et contient différents types d'événements ayant lieu entre deux pulsations consécutives. Ces événements peuvent être des séquences musicales sous format MIDI - fragments mélodiques ou d'accompagnement - ou des données symboliques comme des labels d'accords. Lors des phases de génération, ces différents éléments constituant les états seront tantôt considérés comme des résultats élémentaires à concaténer, tantôt comme des étiquettes (ou labels) à comparer avec la séquence donnée en entrée et servant de chemin à suivre pour le parcours d'un oracle.

3.2. Ecouter et improviser avec l'*oracle live*

ImproteK improvise en recherchant puis réagençant des unités élémentaires pré-éxistantes : les nouvelles phrases sont composées en concaténant des tranches de pulsations provenant de sa mémoire musicale. Ces fragments sont collectés de manière continue en suivant le guide que constitue la grille d'accords servant de référence à la session d'improvisation. Ce processus fait apparaître une première instance de la structure d'oracle : l'*oracle live*, qui effectue son apprentissage sur les phrases jouées par les musiciens (figure 3). Ces entrées MIDI sont en effet indexées pulsation après pulsation en temps réel par les labels d'accords de la grille harmonique en cours (voir section 4.2) et sont donc formatées pour la construction de cet objet.

Une "improvisation" du système correspond à la création d'une nouvelle phrase musicale à partir de la mémoire stockée dans cet oracle live. Dans ce cadre, les labels d'accords de l'oracle sont considérés comme des index pour l'heuristique de *navigation contrainte* (G. Assayag). Ce mécanisme revient à chercher les motifs d'une séquence d'entrée (ici la grille harmonique) dans une séquence différente (ici les labels d'accords des états de l'oracle). Quand un label correspond, la tranche de pulsation mélodique associée est retournée pour être ajoutée à la séquence des

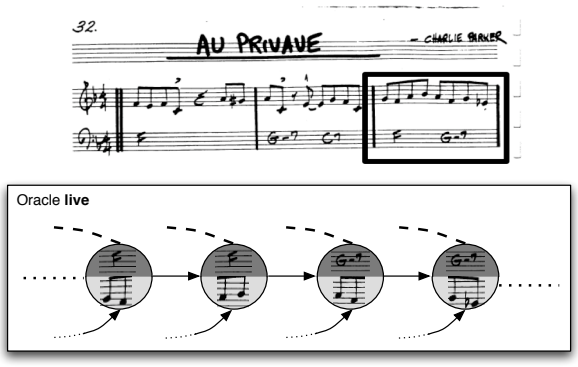


Figure 3. Apprentissage de l'*oracle live*.

résultats des étapes précédentes.

3.3. Navigation contrainte et continuité

Improviser en suivant une grille équivaut ici à suivre un chemin dans l'automate avec des transitions étiquetées par les labels de cette grille. Si aucune transition indexée par le label souhaité n'est trouvée, on cherche à suivre un lien suffixiel permettant d'aller à un autre état où l'on pourra éventuellement lire l'étiquette courante. Les liens suffixiels garantissent qu'il existe, dans la séquence d'origine, une partie commune entre les deux fragments concaténés. La navigation recherche donc en premier lieu la continuité en essayant de coller aux enchaînements appris, et cherche les labels indépendamment s'ils s'inscrivent dans une succession ne figurant à aucun endroit de l'oracle construit sur la séquence d'origine.

Un *paramètre de continuité* est contrôlé à chaque étape du calcul. Il compte le nombre de transitions successives suivies jusqu'alors dans la navigation, et indique ainsi la longueur des segments recopiés de la séquence de l'oracle. En imposant une continuité maximum, on peut donc quantifier l'équilibre souhaité entre la fidélité à la séquence d'origine et l'originalité de la nouvelle improvisation proposée : d'une valeur élevée ressortira une forte similitude, tandis qu'une valeur basse sera à l'origine de plus de surprises.

Lors de la navigation contrainte, les labels successifs de la grille sont lus pour trouver des pulsations indexées par les mêmes labels dans l'oracle live. Si la valeur courante du paramètre de continuité n'excède pas le maximum imposé, la recherche est effectuée en suivant des modes hiérarchisés :

- *Mode "continuity"* : Si son label correspond, suivre une transition, mettre à jour la position de la pulsation courante dans l'oracle, et retourner le fragment mélodique associé. Sinon, passer en *mode suffix* si des liens suffixiels efficaces sont trouvés, ou en *mode nothing* dans le cas contraire.
- *Mode "suffix"* : Suivre le lien suffixiel pointant sur le plus long suffixe répété pour atteindre ensuite un label correspondant (voire figure 4), mettre à jour la position de la pulsation courante dans l'oracle, et re-

tourner le fragment de mélodie associé. Sinon, passer en *mode nothing*.

– *Mode "nothing"* : La recherche de motif est effectuée indépendamment du contexte et le label est cherché dans tout l'oracle. Une transposition peut être opérée si nécessaire.

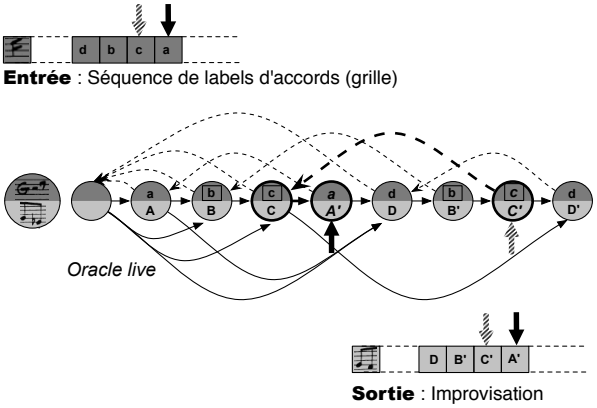


Figure 4. Navigation contrainte à travers l'oracle live.

L'exemple donné par la figure 4 illustre une étape de la navigation. A ce stade, les labels d'accords *d*, *b*, et *c* ont été cherchés et trouvés dans l'oracle live, et la concaténation des fragments musicaux associés *D*, *B´* et *C´* forme la phrase en construction. La position courante dans l'oracle est l'avant-dernier état (*c / C´*) et le label d'accord suivant lu dans la grille donnée en entrée est *a*. Aucune transition pointant sur un label correspondant ne pouvant être trouvé, la recherche continue en *mode suffix*. Le lien suffixiel partant de l'état courant pointe, par définition, sur l'état final du suffixe de *dbc* répété le plus à gauche (*bc*), soit l'état (*c / C*). Une transition correspondant au label *a* recherché est trouvée dans cet état. Le lien suffixiel est donc finalement suivi pour sauter de (*c / C´*) à (*c / C*), ou l'on arrive dans le contexte commun (*bc*) pour pouvoir finalement atteindre la nouvelle pulsation (*a / A´*).

Cette notion de contexte commun est essentielle puisqu'elle assure la cohérence et la musicalité des transitions dans les séquences produites même si la navigation implique des sauts entre différentes zones de l'oracle. En effet, dans le cas de cet exemple, on remarque que ni la séquence d'accords donnée en entrée ni la phrase musicale en sortie ne se trouvaient inscrits en l'état dans la mémoire de l'oracle live.

4. VERS UN INSTRUMENT INTERACTIF

4.1. Architecture

Le jeu des musiciens est reçu, segmenté, formaté et sauvegardé en temps réel via une nouvelle interface développée sous Max/MSP. A travers cette même interface, l'utilisateur d'ImproteK sélectionne une grille d'accords symbolique qui constituera la colonne vertébrale de l'improvisation. Il a également accès aux paramètres de contrôle guidant le calcul des différentes formes de phrases musicales pouvant être générées : de nouvelles improvisations accompagnées ou non (voir section 5) ou des arrangements "live" de la grille (voir paragraphe 4.5).

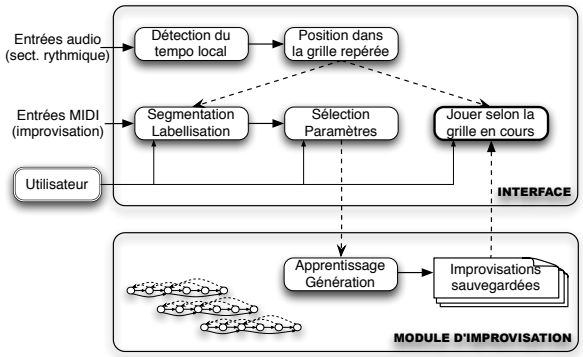


Figure 5. Architecure

Ces instructions de calcul sont envoyées via le protocole OSC au module d'improvisation développé sous OpenMusic présenté dans la section précédente (figure 5), suivant ainsi l'architecture générale d'OMax 2.0 qui fonctionnait en interaction avec Max/MSP grâce à une interface basique ne prenant pas en charge la gestion du tempo développée dans le paragraphe suivant.

Sous cette version, une pulsation métronomique était imposée par la machine, cette condition impliquant nécessairement la perte d'une partie de la richesse musicale et contraignant l'expression des musiciens. Suite à ce constat, un module de suivi de pulsation a été implémenté afin de pouvoir extraire directement le tempo de la session d'improvisation en cours.

4.2. "Get rhythm"

Pendant la performance, une source de pulsation extérieure est nécessaire pour segmenter les entrées provenant des musiciens dans le processus d'écoute, ainsi que pour jouer des phrases rythmiquement en place avec l'improvisation en cours. Pour ce faire, ImproteK fait intervenir l'association d'un module de suivi de pulsation et d'un système de suivi de partition jouant le rôle d'un séquenceur en émettant en temps réel la position courante dans la grille harmonique. Cette donnée sert dans un premier temps à marquer chaque pulsation de l'improvisation live écoutée avec le label d'accord courant. D'autre part, les phrases produites par le module d'improvisation étant des séquences labélisées, ce signal déclenche le jeu des tranches de pulsations correspondantes.

ImproteK se destinant à trouver sa place au sein d'une formation musicale, il était fondamental que son utilisation n'ampute pas la créativité et l'expressivité en subordonnant les musiciens au tempo métronomique imposé par ses sorties. L'objet Max/MSP *beat tracker* [8] a été spécialement développé dans l'optique d'être intégré à un environnement parent d'OMax pour atteindre une musi-

calité plus riche, les expériences antérieures de l'oracle en *mode beat* impliquant un tempo fixe dicté par le système.

Ce module traite des flux MIDI ou audio (par exemple la section rythmique de la formation) et estime en continu le tempo local qui sera utilisé pour générer et jouer des improvisations ajustées à la pulsation humaine nécessairement variable. Pendant une phase d'initialisation, l'utilisateur fournit une indication de tempo par une battue manuelle dont le but est également de donner la phase initiale en indiquant les dates absolues des pulsations pour éviter ainsi une synchronisation à contre-temps. Après cette étape, les variations de tempo sont calculées tout au long de la performance (la description de cette unité, n'étant pas le sujet du présent article, voir [8] pour plus de détails). Ce signal est ensuite utilisé comme une horloge venant déclencher les différents éléments constituant les séquences chargées dans Antescofo.

4.3. Utiliser un *score follower* comme séquenceur

Antescofo [12] est un système de suivi de partitions polyphoniques et un langage de programmation synchrone pour la composition musicale conçu par A. Cont. Cet objet, développé en tant que module externe pour Max/MSP, procède à la reconnaissance automatique de la position et du tempo dans une partition musicale à partir d'un flux audio provenant d'un ou plusieurs interprètes. Il permet ainsi la synchronisation de la performance instrumentale avec des éléments informatiques inscrits dans une partition électronique.

Chaque séquence calculée par le module d'improvisation et destinée à être restituée est donc écrite et sauvegardée sous le format d'une partition Antescofo, pouvant ainsi être jouée à partir de la position courante de la grille dès son chargement. Les phrases ainsi générées viennent enrichir une collection constituée au fil des sessions pour pouvoir être chargées par l'utilisateur au cours de l'improvisation. Dans notre utilisation d'Antescofo, ce sont les pulsations fournies par l'objet *beat tracker* qui tiennent lieu de "notes" dans la partition. Elles viennent déclencher les évènements que sont les ordres de jouer les tranches MIDI contenues entre deux pulsations. La notation temporelle dans une partition Antescofo pouvant s'effectuer en temps relatif, il est donc possible de faire jouer à un tempo donné une phrase générée à partir d'un matériau de base dont l'acquisition avait été faite à un tempo différent.

4.4. Segmenter et indexer les entrées

Ce couple d'objets en charge de la gestion de la pulsation se trouve en aval pour jouer les séquences générées, mais également en amont dans le processus d'acquisition en temps réel. Au début de chaque session d'improvisation, on se place dans le contexte d'une grille harmonique connue. Celle-ci est écrite sous la forme d'une suite de labels d'accords correspondant chacun à une pulsation et est sauvegardée sous le format Antescofo avec un encodage particulier : on utilise le canal MIDI numéro 16 pour représenter la grille avec des conventions permettant d'exprimer la fondamentale de l'accord et sa nature.

Lors de l'acquisition des entrées, ces informations sont envoyées par Antescofo et figurent donc dans les buffers MIDI dans lesquels viennent lire les fonctions de la bibliothèque OpenMusic. Ces annotations permettent de segmenter les séquences MIDI par pulsations et d'attribuer à chacune d'elles un label harmonique dans l'optique de la construction d'oracles en *mode beat* ou de séquences musicales.

4.5. Un instrument force de propositions

ImproteK est un instrument joué par un interprète interagissant avec le reste de la formation au même titre qu'un instrumentiste traditionnel. Ce dernier pilote le système à l'aide de l'interface graphique pour contrôler en temps réel les paramètres des étapes d'apprentissage comme de génération qui sont initiées sur demande. Pendant la performance, il sélectionne les phrases musicales à donner en entrée pour la modélisation stylistique, choisit les oracles courants (l'oracle live tout comme les autres instances impliquées dans le module d'harmonisation et d'arrangement décrit dans la section 5) et il détermine les paramètres techniques comme la continuité maximum pour la navigation dans chaque oracle. Entre autres caractéristiques, le terrain d'apprentissage, la longueur ou même "l'audace" des improvisations sont donc laissées à sa discrétion.

Cet interprète n'est pas uniquement en charge de l'écoute et de l'apprentissage : il dirige pleinement le jeu au moyen de contrôles au clavier. Il peut accéder aux phrases créées à partir des derniers évènements musicaux entendus comme à celles issues des sessions précédentes. Différentes formes d'improvisations lui sont disponibles : elles peuvent être par exemple des phrases mélodiques accompagnées ou non, ou des arrangements live de la grille. Grâce au format d'écriture des partitions Antescofo et à la connaissance de la position courante dans la grille, chaque phrase lancée se met immédiatement à jouer à partir de cette position. Il est donc ainsi possible de changer la phrase en cours de jeu pendant un seul et même passage de la grille et par exemple de naviguer entre plusieurs improvisations pour créer une improvisation hybride.

Enfin, le rôle joué par ImproteK peut évoluer pendant une même session : il peut tout aussi bien être soliste et/ou accompagnateur selon la nature des séquences choisies par l'interprète pour remplir l'oracle live. En effet, faire effectuer à cet oracle son apprentissage sur le jeu d'un accompagnateur permettra de réaliser ensuite un arrangement de la grille en temps réel. L'autre approche disponible de la question de l'accompagnement est celle fournie par le module d'harmonisation et d'arrangement présenté dans la section suivante.

5. IMPROVISATIONS HARMONISÉES ET ARRANGÉES

Ce module d'harmonisation et d'arrangement peut être utilisé de manière autonome comme une entité indépendante créant un accompagnement pour une mélodie sans interaction temps réel. Cependant, il a été pensé pour être intégré dans l'environnement plus vaste d'ImproteK afin de composer de nouvelles improvisations accompagnées.

Il ne connaît ni applique aucune règle musicale et est exclusivement fondé sur l'utilisation d'un corpus qui est simultanément considéré comme un terrain d'apprentissage pour extraire des mécanismes empiriques, et comme une mémoire musicale dans laquelle piocher pour concrétiser les accompagnements après qu'ils ont été calculés.

5.1. L'apprentissage

5.1.1. Le corpus

L'apprentissage du corpus est effectué sur trois éléments : la *piste mélodique* (thème, improvisation, solo, etc.), la *piste d'accompagnement*, et la *grille harmonique associée*. Il n'est pas conduit sur des données purement symboliques comme des partitions mais sur les performances en direct, et est actuellement constitué de standards de jazz et de morceaux de Bernard Lubat.

On emploiera le terme "corpus" pour désigner l'ensemble des modèles, qu'ils soient extraits d'un apprentissage différé ou construits lors de la performance en cours. Ici encore, le système écoute les musiciens, segmente ses entrées par pulsation, et apprend ces séquences ainsi que les associations entre ces trois éléments (mélodie, grille, accompagnement) en construisant des instances d'oracle. La phase d'apprentissage est en réalité double : un couple d'oracles est construit pour chaque élément du corpus : un oracle d'harmonisation et un oracle d'arrangement (figure 6).

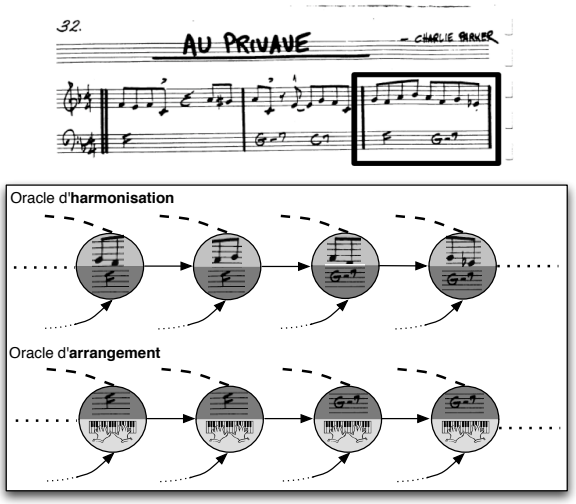


Figure 6. Apprentissage des oracles d'arrangement et d'harmonisation

Un *oracle d'harmonisation* mémorise les séquences d'associations entre les fragments mélodiques et les labels d'accords pour les pulsations qu'il couvre. Dans le cas d'un *oracle d'arrangement*, les associations apprises sont celles entre les labels d'accords et les fragments d'accompagnement. Comme leurs noms l'indiquent, ces deux objets seront respectivement utilisés pour associer une progression harmonique symbolique à une mélodie, et un accompagnement à une progression harmonique symbolique.

5.1.2. Classes d'équivalence

Cette construction incrémentale tout comme les filtrages par motif successifs décrits dans le prochain paragraphe (5.2.1) est en pratique exécutée sur des classes d'équivalence. Pour un oracle d'harmonisation, deux états sont considérés comme équivalents s'ils sont indexés par les mêmes notes sans prendre en compte leurs durées, leurs répétitions ou leur ordre dans la tranche de pulsation. De la même manière, deux états d'un oracle d'arrangement sont équivalents s'ils sont indexés par les mêmes labels d'accords, quels que soient les fragments d'accompagnement associés.

Ces équivalences ont été introduites pour éviter la rigidité improductive qu'aurait pu amener un critère d'égalité stricte. Les conséquences structurelles pour les oracles concernent le nombre de liens suffixiels efficaces comme l'illustre l'exemple simplifié de la figure 7 appliqué à une chaîne de caractère : un oracle construit sur le mot *oracle*.

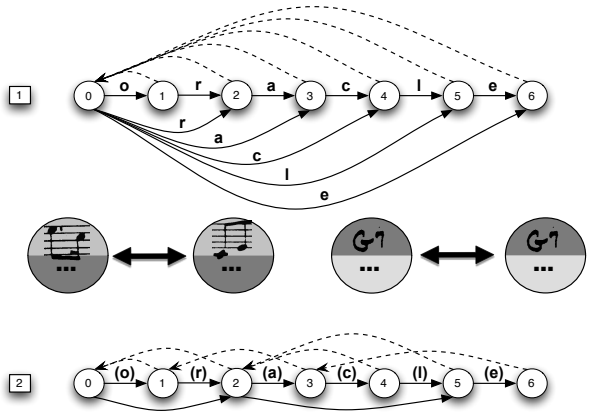


Figure 7. Classes d'équivalence et liens suffixiels.

Dans le premier exemple, chaque lien suffixiel pointe par convention sur l'état initial car aucun motif répété n'est reperé (chaque lettre n'apparaissant qu'une fois). Dans le deuxième cas introduisant les classes d'équivalence *voyelles* et *consonnes*, des motifs répétés sont observés et la structure est plus complexe.

5.2. Harmonisation et arrangement

5.2.1. Cascade d'oracles

Harmonisation et arrangement sont réalisés en cascade : une navigation contrainte est effectuée dans un premier

temps à travers un oracle d'harmonisation, puis à travers un oracle d'arrangement (figure 8).

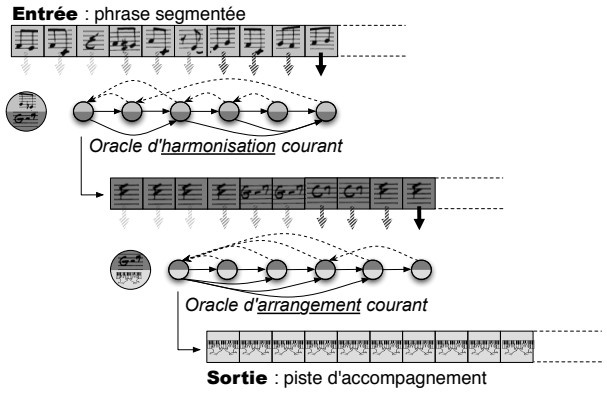


Figure 8. Harmonisation et arrangement en cascade.

L'étape d'harmonisation retourne une progression de labels d'accords, et cette séquence symbolique devient elle-même le chemin à suivre pour la navigation dans l'oracle d'arrangement. La recherche de motifs est cette fois exécutée sur les étiquettes que constituent ces labels d'accords pour obtenir finalement une séquence formée par la concaténation des tranches d'accompagnement trouvées à chaque étape de la recherche.

La continuité intrinsèque à l'utilisation de la structure d'oracle apporte des progressions harmoniques cohérentes pour la phase d'harmonisation. Elle propose également une reconstruction homogène à partir des différentes pulsations du corpus assemblées pour créer le nouvel accompagnement. Ces propriétés de continuité se répercutent par exemple sur les lignes de basses dans les accompagnements complexes évitant ainsi des phrasés trop accidentés.

5.2.2. Intermédiaire formel

Il est important de souligner que, même si une étape intermédiaire implique des données symboliques comme des labels d'accords, aucune règle harmonique n'est appelée pour procéder à l'harmonisation et à l'arrangement. La dénomination choisie pour ce langage formel intermédiaire est seulement destinée à l'utilisateur pour rendre la performance plus intuitive : le système lui-même ignore la signification musicale de ce marquage et ne considère deux labels d'accords différents que comme deux index à comparer.

L'insertion de ce langage formel à un niveau intermédiaire séparant le procédé en deux étapes disjointes est motivée par trois raisons. Tout d'abord, il découle naturellement de la notation usuelle des standards de jazz comme on peut par exemple les trouver dans les Realbooks où un thème est mis en regard d'une suite d'accords correspondante. De plus, cette factorisation du problème permet de multiplier les possibilités : une phrase peut en effet être harmonisée à l'aide d'un oracle donné et ensuite arrangée avec un oracle d'arrangement appris sur un tout autre corpus. Enfin, cet intermédiaire permettra d'implémenter dans un développement futur un niveau optionnel de substitutions basé sur une grammaire [10].

Les notations et le vocabulaire employés ne doivent pas masquer la portée générale de ce module. En effet, les termes "harmonisation" et "arrangement" viennent des situations dans lesquelles ImproteK a été utilisé jusqu'à maintenant : des sessions d'improvisation dans un contexte de jazz tonal. Dans d'autres environnements musicaux, sa généricité lui permet de comprendre d'autres formes d'associations verticales pouvant être indexées de manière agnostique en employant une autre grammaire.

6. EXPÉRIMENTATIONS ET RÉSULTATS

ImproteK a servi de partenaire virtuel à des musiciens professionnels, en particulier Bernard Lubat lors de séances d'improvisation menées à Uzeste depuis 2011. Des exemples audio et vidéo illustrant les différents points abordés dans l'article sont disponibles sur la page :
http ://ehess.modelisationsavoirs.fr/improtech/improtek.

Parmi eux, le contrôle du logiciel en temps réel à l'aide de l'interface Max/MSP est présenté par le biais d'une improvisation basée sur des transcriptions du style mambo d'Erroll Garner. L'interaction directe entre la machine et un instrumentiste est illustrée par des extraits de sessions où le musicien, Bernard Lubat, accompagne la machine jouant une improvisation inspirée de son propre chorus, suivis d'une discussion sur le résultat.

Une autre série d'exemples expose l'éventail de résultats possibles pour le module d'harmonisation et d'arrangement en proposant successivement plusieurs accompagnements pour une même improvisation jouée en boucle. En effet, selon les choix de l'utilisateur concernant les différentes parties du corpus à utiliser et les paramètres comme la continuité imposée, le rendu peut aller de la simple imitation lorsque les oracles d'harmonisation, d'arrangement, et live sont choisis dans une même partie du corpus, à l'originalité voire l'extravagance quand des oracles complètement indépendants ont été activés.

7. CONCLUSION

Le système d'improvisation musicale présenté dans cet article est capable de dégager les logiques des associations horizontales et verticales sous-tendant une performance d'improvisation pour devenir lui-même force de proposition en développant son esthétique propre inspirée de celle de ses partenaires. Sa matière première est en effet leur jeu : il est simultanément employé comme terrain d'apprentissage pour la modélisation du style et comme mémoire musicale pour concrétiser ses propres improvisations.

Les propriétés de l'oracle structurant sa mémoire permettent de s'affranchir du dilemme du choix entre innovation et cohérence en assurant la continuité par construction, et en permettant ainsi de travailler avec le grain fin qu'est la pulsation. Cette unité est choisie comme élément de base pour les calculs comme pour la restitution finale

et est servie par un module de suivi de pulsation pour une interaction enrichie.

ImproteK est en effet conçu comme un instrument à part entière et nécessite un interprète pour diriger l'apprentissage, la génération, et le jeu en temps réel. Il peut tour à tour être soliste ou accompagnateur, et peut également proposer des improvisations accompagnées grâce au module d'harmonisation et d'arrangement.

A plus court terme que certains objectifs techniques comme la prise en charge de l'audio, les directions prises actuellement par le projet ImproteK visent à évaluer finement la compatibilité entre la progression harmonique de la grille de la session d'improvisation et celles résultant du module d'harmonisation et d'arrangement. Au stade actuel, l'interprète observe la comparaison entre les deux grilles par l'intermédiaire de l'interface affichant les labels d'accords respectifs après chaque création d'improvisation accompagnée. Cette étude permettra une meilleure intégration de l'interaction harmonique dans l'instrument, et rendra son utilisation plus intuitive.

8. REMERCIEMENTS

Avec le soutien de l'ANR.
Projet IMPROTECH ANR-09-SSOC-068.

Nous souhaitons remercier la famille OMax, Gérard Assayag, Georges Bloch, et Benjamin Lévy pour les échanges fructueux d'expériences et d'idées concernant la conception et l'implémentation d'ImproteK. Nous remercions également Laurent Bonnasse-Gahot qui lui a donné le sens du rythme avec le *beat tracker*, ainsi que Carlos Agon et Jean Bresson pour leurs conseils concernant OpenMusic, et Arshia Cont pour ses modifications d'Antescofo sur mesure. Nous adressons enfin un remerciement tout particulier à *La Compagnie Lubat* pour ses accueils répétés et les collaborations toujours plus enrichissantes.

9. REFERENCES

[1] C. Allauzen, M. Crochemore, and M. Raffinot, "Factor oracle : A new structure for pattern matching," in *SOFSEM 99 : Theory and Practice of Informatics*. Springer, 1999, pp. 758–758.

[2] G. Assayag and G. Bloch, "Navigating the oracle : A heuristic approach," in *International Computer Music Conference*, vol. 7, 2007, pp. 405–412.

[3] G. Assayag, G. Bloch, and M. Chemillier, "Omaxofon," *Sound and Music Computing (SMC)*, 2006.

[4] G. Assayag, G. Bloch, M. Chemillier, A. Cont, and S. Dubnov, "Omax brothers : a dynamic topology of agents for improvization learning," in *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*. ACM, 2006, pp. 125–132.

[5] G. Assayag and S. Dubnov, "Using factor oracles for machine improvisation," *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, vol. 8, no. 9, pp. 604–610, 2004.

[6] G. Assayag, S. Dubnov, and O. Delerue, "Guessing the composerÕs mind : Applying universal prediction to musical style," in *Proceedings of the International Computer Music Conference*, 1999, pp. 496–499.

[7] J. Biles, "Genjam : Evolutionary computation gets a gig," in *Proceedings of the 2002 Conference for Information Technology Curriculum, Rochester, New York, Society for Information Technology Education*, 2002.

[8] L. Bonnasse-Gahot, "Donner à omax le sens du rythme : vers une improvisation plus riche avec la machine," École des Hautes Études en sciences sociales, Tech. Rep., 2010, http ://ehess.modelisationsavoirs.fr/improtech/docs.

[9] J. Bresson, C. Agon, and G. Assayag, "Openmusic 5 : A cross-platform release of the computer-assisted composition environment," in *10th Brazilian Symposium on Computer Music, Belo Horizonte, MG, Brésil*, 2005.

[10] M. Chemillier, "Toward a formal study of jazz chord sequences generated by steedman´ s grammar," *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, vol. 8, no. 9, pp. 617–622, 2004.

[11] C. Chuan and E. Chew, "A hybrid system for automatic generation of style-specific accompaniment," in *4th Intl Joint Workshop on Computational Creativity*, 2007.

[12] A. Cont, "Antescofo : Anticipatory synchronization and control of interactive parameters in computer music," in *Proceedings of the International Computer Music Conference*, 2008.

[13] B. Lévy, "Visualising omax," Master's thesis, Master ATIAM, Université Pierre et Marie Curie, Paris VI - IRCAM, 2009.

[14] G. Lewis, "Too many notes : Computers, complexity and culture in voyager," *Leonardo Music Journal*, pp. 33–39, 2000.

[15] J. Nika, "Intégrer l'harmonie dans un processus informatique d'improvisation musicale," Master's thesis, Master ATIAM, Université Pierre et Marie Curie, Paris VI - IRCAM, 2011, http ://articles.ircam.fr/textes/Nika11a/index.pdf.

[16] F. Pachet, "The continuator : Musical interaction with style," *Journal of New Music Research*, vol. 32, no. 3, pp. 333–341, 2003.

[17] G. Ramalho, P. Rolland, and J. Ganascia, "An artificially intelligent jazz performer," *Journal of New Music Research*, vol. 28, no. 2, pp. 105–129, 1999.

[18] R. Rowe, *Interactive music systems : machine listening and composing*. MIT press, 1992.

[19] I. Simon, D. Morris, and S. Basu, "MySong : automatic accompaniment generation for vocal melodies," in *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*. ACM, 2008, pp. 725–734.

[20] B. Thom, "BoB : an interactive improvisational music companion," in *Proceedings of the fourth international conference on Autonomous agents*. Citeseer, 2000, pp. 309–316.

[21] D. Zicarelli, "M and jam factory," *Computer Music Journal*, vol. 11, no. 4, pp. 13–29, 1987.

L'AMBISONIE D'ORDRE SUPERIEUR ET SON APPROPRIATION PAR LES MUSICIENS : PRESENTATION DE LA BIBLIOTHEQUE MAX/MSP HOA.LIB

Julien Colafrancesco
CICM - Centre de recherche en Informatique et Création Musicale
Université de Paris 8
Maison des Sciences de l'Homme, Paris Nord
jcolafrancesco@gmail.com

RÉSUMÉ

Cet article a pour but de présenter un ensemble d'outils informatiques dédiés à la spatialisation ambisonique du son. Ces derniers sont accessibles sous forme d'objets MAX/MSP et visent des applications musicales expérimentales. A travers une brève présentation des divers modèles de restitution sonore basés sur la décomposition du champ acoustique en harmoniques sphériques, nous exposerons certains concepts clefs dans le cadre d'une appropriation musicale de l'ambisonie. Nous présenterons par la suite la bibliothèque d'objets et le modèle modulaire d'après lequel elle a été conçue

1. INTRODUCTION

L'espace sonore est une des principales dimensions de la pensée musicale contemporaine, spécialement dans le domaine de la musique électroacoustique, mais également dans les arts intermédia. C'est dans ce contexte que le CICM a fait de la mise en espace des sons l'un de ses principaux axes de recherche [1] [2] , en proposant notamment des outils favorisant, pour le musicien, l'expérimentation artistique ainsi que l'appropriation des concepts théoriques liés à la spatialisation. Ainsi, dès 2004 notre équipe a développé les « CICM Tools », une bibliothèque pour MAX/MSP et Pure Data offrant une spatialisation en VBAP ainsi qu'en ambisonique de format B. Plus récemment dans le cadre du projet HD3D2, le « CICM Panner » [1] , un plugin VST de panoramique d'amplitude destiné à l'industrie audiovisuelle, a été développé.

Les outils présentés dans cet article visent particulièrement l'approche ambisonique en en proposant une implémentation plus avancée que dans le cadre de nos précédents travaux. Basée sur la thèse de Jérôme Daniel, cette bibliothèque se désolidarise en effet du format B (ambisonique d'ordre 1) et donne accès à un modèle d'ordre supérieur. Une prise en compte des effets de distance est aussi proposée, cette dernière nous permettant de modeler la courbure du front d'onde synthétisé.

Afin de saisir l'ensemble des enjeux liés à de telles modifications, la première partie de cet article tentera d'introduire la théorie relative aux techniques ambisoniques. Il s'agit d'un exposé synthétique à destination d'un public non-initié. Nous introduirons ainsi la décomposition du champ acoustique en harmoniques sphériques et les principaux modèles de sources sonores (l'onde plane et l'onde sphérique). Ces quelques bases nous permettront de mettre en valeur les principales différences entre la conception standard de l'ambisonie [3] et celle proposée par Jérôme Daniel [4].

Par la suite seront évoqués certains détails d'implantation. Nous avons, à la manière de Graham Wakefield [5], opté pour une organisation modulaire de notre bibliothèque. L'utilisateur peut ainsi composer son moteur ambisonique en connectant un ensemble d'objets. Cette organisation permet de faire face à diverses situations. De plus, elle laisse au musicien l'accès à un grand nombre de degrés de liberté dont il pourra potentiellement s'emparer à des fins artistiques.

Dans cette optique, le dernier axe de cet article comportera une discussion ouverte sur l'ambisonie et sa possible appropriation par les musiciens via notre suite d'outils, ceci en proposant quelques exemples concrets d'opérations musicales à effectuer dans le domaine des harmoniques sphériques.

2. LES FONDEMENTS ACOUSTIQUES DE L'AMBISONIE

L'ambisonie est basée sur une représentation physique du champ sonore située dans un domaine dit aux harmoniques sphériques. La représentation dont nous parlons décompose un champ en une somme pondérée de fonctions spatiales. Tout champ sonore peut ainsi être codé efficacement par une série de

1 . Les CICM-Tools et le CICM-Panner sont librement téléchargeables sur le site http ://cicm.mshparisnord.org/.

coefficients de pondération.

2.1. Description du champ sonore dans le domaine des harmoniques sphériques, le cas du problème intérieur

Le problème intérieur peut être représenté par la figure 1. Dans le cadre de ce problème, les sources sonores sont situées à l'extérieur d'une sphère de rayon r_{max}. Il s'agit d'un modèle assez proche du cas pratique en jeu dans cet article où nous cherchons à contrôler le champ sonore dans une zone sphérique délimitée par un ensemble de haut-parleurs.

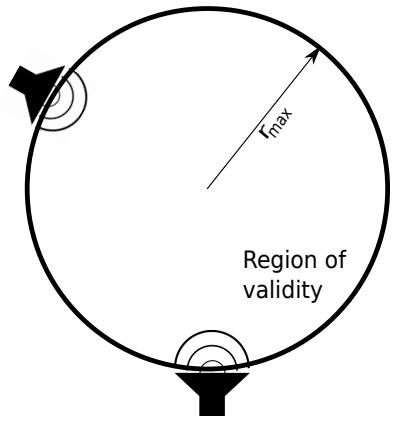


Figure 1. Représentation du problème intérieur

Dans le cas du problème intérieur, une résolution de l'équation de Helmholtz [2] en coordonnées sphériques nous donne l'équation

$$p(r,\theta,\phi,k) = \sum_{n=0}^{\infty}\sum_{m=-n}^{n} b_n^m(k) j_n(kr) Y_n^m(\theta,\phi)\,, \qquad (1)$$

avec (θ,ϕ,r) les coordonnées sphériques, k le nombre d'onde, p la pression, Y_n^m les harmoniques sphériques d'ordre n et de degré m, j_n les fonctions de Bessel sphériques et b_n^m les coefficients de la décomposition. Pour de plus amples informations sur la manière dont cette équation est dérivée et sur la nature des différentes fonctions en jeu, le lecteur intéressé peut se référer aux ouvrages [6], [7] et [8]. Dans le cadre de cet article, l'important est de noter qu'un champ sonore peut-être totalement représenté via une suite de coefficients b_n^m.

Cette décomposition du champ sonore en harmoniques sphériques peut être vue de manière analogue à une décomposition en série de Fourrier où une fonction périodique est décomposée en une somme pondérée de fonctions sinusoïdales. Dans une décomposition de Fourier, la reconstitution est d'autant plus fine que le nombre de composantes fréquentielles augmente. Il en va de même pour une représentation dans le domaine des harmoniques sphériques du champ sonore : plus nous considérons de composantes spatiales d'ordre élevées, plus le champ sonore est représenté en détails. Cette remarque est particulièrement importante dans le cadre de l'ambisonie, où, comme nous allons bientôt le voir, la décomposition du champ sonore est tronquée à un ordre N (i.e. que les composantes spatiales d'ordre supérieur à N ne sont pas considérées).

2.2. Les différents modèles de source sonore

Le principe de superposition nous indique qu'un champ acoustique peut être décomposé en une somme de champs relatifs à chaque source sonore en présence. Nous pouvons ainsi associer à une source sonore un jeu de coefficients b_n^m.

En acoustique, deux modèles de source sont principalement utilisés, la source sonore ponctuelle et l'onde plane. La source sonore ponctuelle est une source située en un point précis de l'espace. Son front d'onde est de nature sphérique, la courbure et l'amplitude de ce dernier se réduisant au fur et à mesure de la propagation. L'onde plane est quant à elle issue d'une source sonore infiniment éloignée, elle est d'amplitude constante et ne présente pas de courbure.

Via les dérivations détaillées en [9], nous avons :

$$b_n^m(k, r_s) = s Y_n^m(\theta_s, \phi_s) l_n(kr_s) \qquad (2)$$

avec

$$l_n(kr_s) = \begin{cases} 4\pi i^n & \text{pour une onde plane,} \\ -ikh_n^{(2)}(kr_s) & \text{pour une onde sphérique,} \end{cases}$$

(θ_s, ϕ_s, r_s) les coordonnées sphériques de la source sonore, $i = \sqrt{-1}$, $h_n^{(2)}$ la fonction de Hankel sphérique de seconde espèce et s le signal de la source sonore.

Nous pouvons ainsi associer à chaque source sonore une série de coefficients spatiaux. Dans le cadre d'un dispositif de restitution ambisonique, il pourrait s'agir de la source sonore virtuelle à reproduire aussi bien que d'un des hauts parleurs physiquement présents.

3. L'AMBISONIE

L'ambisonie consiste à approcher la série de coefficients relative à la source sonore virtuelle via une combinaison des séries de coefficients relatives à chaque haut-parleur. Cependant, et comme nous avons pu le voir en 2.2, plusieurs modèles de sources sonores existent. Lequel choisissons nous ?

2 . L'équation de Helmholtz peut être dérivée à partir de l'équation des ondes. Il s'agit d'une équation différentielle gouvernant la propagation des ondes acoustiques en milieu stationnaire (e.g. l'air).

3.1. Le modèle ambisonique initial

La manière la plus commune de concevoir l'ambisonie est de considérer que la source virtuelle à reproduire ainsi que tous les haut-parleurs sont suffisamment éloignés pour que leurs ondes puissent être considérées comme planes. Le "format B" [3] est un cas particulier de cette conception où les composantes spatiales sont limitées à l'ordre 1.

Ce type de système ambisonique peut-être représenté sous forme matricielle de la manière suivante :

$$C\vec{s} = \vec{b} \tag{3}$$

avec

$$C = \begin{bmatrix} Y_0^0(\vec{\theta_1}) & Y_0^0(\vec{\theta_2}) & \dots & Y_0^0(\vec{\theta_L}) \\ Y_1^{-1}(\vec{\theta_1}) & Y_1^{-1}(\vec{\theta_2}) & \dots & Y_1^{-1}(\vec{\theta_L}) \\ \vdots & \vdots & \vdots & \vdots \\ Y_N^N(\vec{\theta_1}) & Y_N^N(\vec{\theta_2}) & \dots & Y_N^N(\vec{\theta_L}) \end{bmatrix}, \tag{4}$$

$$\vec{s} = [s_1 \quad s_2 \quad \dots \quad s_N]^T, \tag{5}$$

$$\vec{b} = [sY_0^0(\vec{\theta_s}) \quad sY_1^{-1}(\vec{\theta_s}) \quad \dots \quad sY_N^N(\vec{\theta_L})]^T.$$

$\vec{b}$ représente les coefficients de la source sonore virtuelle [3], C l'encodage directionnel de chaque haut-parleur, $\vec{s}$ les signaux émis par les haut-parleurs, N l'ordre de la décomposition et L le nombre de haut-parleurs.

En calculant la matrice $C^\dagger$,la pseudo-inverse de C, nous pouvons obtenir les signaux $\vec{s}$, en effet :

$$\vec{s} = C^\dagger \vec{b}. \tag{6}$$

Les coefficients b_n^m dans le cas de l'onde plane n'ayant pas de dépendance fréquentielle et étant de plus réels ou imaginaires purs (en fonction de leur ordre), une telle conception de l'ambisonie nous permettra d'approcher les coefficients relatifs à la source sonore virtuelle en appliquant un simple gain à chaque haut-parleur. On notera qu'une résolution de ce système implique un nombre de haut-parleurs au moins égal au nombre de composantes spatiales [4].

3.2. compensation du champ proche

La conception précédente bénéficie d'une certaine simplicité, l'ambisonie pouvant être considérée comme un cas de panoramique d'amplitude. Deux problèmes se posent pourtant, ils sont dus au choix du modèle de l'onde plane pour représenter les contributions des haut-parleurs et de la source sonore virtuelle. Les haut-parleurs sont en effet davantage assimilables à des sources sonores ponctuelles (ils ne sont pas infiniment éloignés de l'auditeur). Négliger cet aspect conduit à des approximations dans la reproduction du champ sonore. Pour ce qui est du front d'onde à synthétiser, choisir le modèle de l'onde plane constitue une réduction en terme de degrés de liberté, il devient impossible de synthétiser le champ sonore relatif à la présence d'une source sonore ponctuelle. Du point de vue de l'auditeur, il y a privation des effets de distances perçus via les variations de courbures du front d'onde.

Afin de remédier à ces deux problèmes, Jérôme Daniel propose un nouveau format ambisonique [4] considérant un modèle d'onde sphérique plutôt que plane. La pertinence de ce format vient en grande partie de sa compacité ainsi que de sa compatibilité avec la conception en onde plane. Le dispositif ambisonique pouvant être représenté sous forme matricielle de la manière suivante :

$$\vec{s} = C^\dagger H \vec{b} \tag{7}$$

avec

$$H = \mathrm{Diag}([\dots\ h_n^{\mathrm{NFC}}(k, r_{sc}, r_{hp})\ \dots]), \tag{8}$$

$$h_n^{\mathrm{NFC}}(k, \rho, R) = \frac{l_n(kr_{sc})/l_0(kr_{sc})}{l_n(kr_{hp})/l_0(kr_{hp})}, \tag{9}$$

r_{sc} la distance de la source virtuelle et r_{hp} la distance des haut-parleurs.

4. UNE BIBLIOTHEQUE POUR L'AMBISONIE D'ORDRE SUPERIEUR

La bibliothèque Hoa.Lib propose un ensemble d'objets MAX/MSP destinés à l'ambisonie d'ordre supérieur. Elle s'inspire de la bibliothèque ambisonique de Graham Wakefield [5] dans la mesure où elle reprend son concept d'architecture modulaire. Si les deux outils peuvent être vus de manière analogue en ce qui concerne leur architecture, leurs ambitions et fonctionnalités demeurent sensiblement différentes. Le travail de Graham Wakefield respecte les différentes normes et conventions relatives à l'ambisonie, il y a donc une parfaite compatibilité avec les outils et matériaux ambisoniques pré-existants (e.g. les fichiers audio enregistrés au format B). L'ordre est cependant limité à l'ordre 3 et la prise en compte du champ proche n'est pas proposée.

Hoa.Lib a quant à elle été créée comme un ensemble d'outils autonomes dédiés à l'expérimentation musicale. Elle a été développée via la bibliothèque C++ de calcul matriciel Boost : :Ublas et propose ainsi l'accès à un ordre variable, uniquement limité en pratique par les capacités de calcul de la machine. Hoa.Lib propose de plus une compensation des coefficients afin de prendre en compte les distances auxquelles sont situés les haut-parleurs ainsi

3 . Dans ce système, il est possible de négliger le facteur $4\pi i^n$ des coefficients car constant pour un ordre donné.

4 . Le nombre de composantes spatiales étant égal à $(N+1)^2$ pour un système ambisonique tridimensionnel et $2N+1$ pour un système bidimensionnel

que la source sonore virtuelle. La description théorique de l'ambisonie faite dans la section 3 a donc été implantée dans son ensemble.

4.1. La bibliothèque en détail

La figure 2 représente un patch MAX/MSP exécutant un encodage puis un décodage ambisonique d'ordre 3 à destination de 8 haut-parleurs. Cet exemple est basé sur la conception de l'ambisonie décrite en 3.1 car seuls les modules d'encodage et de décodage sont présents.

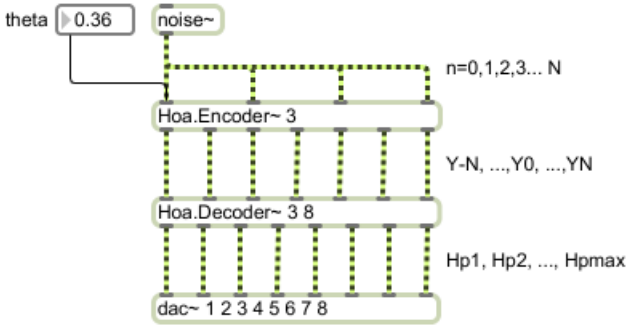


Figure 2. Encodage et décodage basique à l'odre 3

Sur la figure 2 nous pouvons remarquer trois niveaux rendant possible une éventuelle interaction :

- **Post-décodage** : nous avons directement accès aux signaux prêts à être restitués par les haut-parleurs.
- **Pré-décodage** : nous disposons des signaux résultant de l'encodage spatial.
- **Pré-encodage** : nous agissons sur le signal de la source sonore virtuelle.

Pré-encodage, le signal est copié sur un nombre de canaux égal à $N+1$. Ceci permet une factorisation des traitements linéaires à effectuer sur les composantes spatiales en fonction de leur ordre.

Tel que l'indique l'équation (8), la compensation du champ proche peut être appliquée par filtrage des signaux encodés spatialement. Dans le cadre de ce travail, nous avons implanté ces traitements par convolution. En conséquence, les calculs peuvent rapidement se révéler prohibitifs à mesure que le nombre de signaux à considérer augmente.

La figure 3 est un exemple illustrant une factorisation des opérations de filtrage. Comme nous pouvons le constater, ceci a pour effet de réduire drastiquement le nombre d'opérations de convolution à effectuer.

Il existe toutefois certaines situations où il est préférable d'effectuer la compensation post-encodage. Par exemple, si nous souhaitons spatialiser plusieurs sources sonores tout en sachant que ces dernières seront à la même distance du point d'écoute, nous

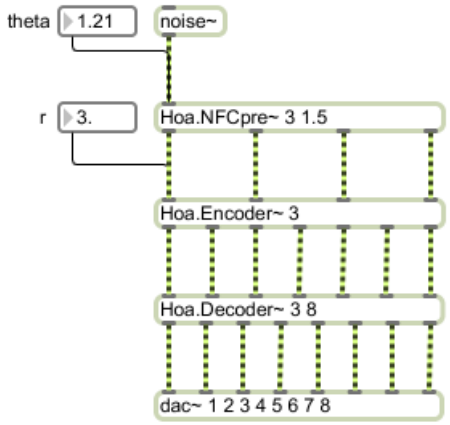


Figure 3. Filtrage NFC pre-encodage

pouvons effectuer un encodage pour chaque source puis sommer les canaux issus de l'encodage de manière à effectuer simultanément la compensation des différents signaux. La figure 4 illustre ce cas, nous pouvons y noter la présence d'un seul et unique module de compensation.

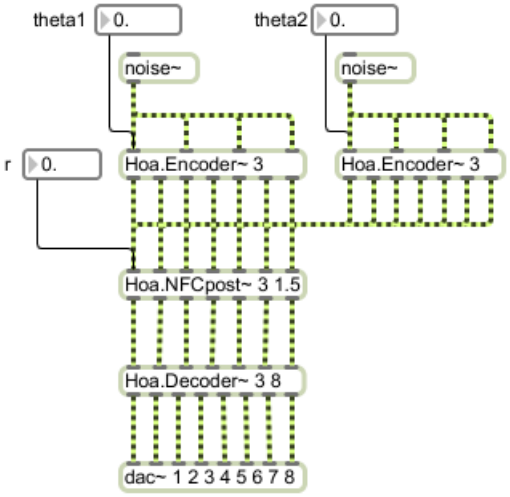


Figure 4. Filtrage post-encodage

4.2. Limitations

La bibliothèque présentée dans cet article est sujette à un certain nombre de limitations. Certaines d'entres elles ne sont que temporaires et devraient être levées lors des prochaines mises à jour. Le système proposé est notamment réduit à un espace à deux dimensions, aucune gestion de l'élévation n'est possible. L'écart angulaire entre les différents haut-parleurs doit de plus être parfaitement régulier. Une configuration irrégulière impliquerait un mauvais conditionnement de la matrice de décodage qui sous-

optimiserait le nombre de haut-parleurs nécessaire pour atteindre un ordre donné. Certaines solutions ont été trouvées afin de répondre à ce problème. En effet, en calculant la pseudo-inverse de la matrice C à l'aide d'une décomposition en valeurs singulières [10], une amélioration des résultats devient possible.

Certaines limitations dépendent davantage du format ambisonique considéré. La conception présentée en 3.2 ne peut gérer les configurations composées de haut-parleurs situés à des distances différentes du point d'écoute. Il est aussi impossible de supporter le fait que plusieurs haut-parleurs puissent être dépendants du même canal. La levée de ces deux limitations ouvrirait la voie à des applications ambisoniques avancées. Nous pourrions par exemple imaginer un système où chaque source sonore miroir, due aux réflexions sur les parois de la salle, serait prise en compte dans la matrice de décodage, il s'agirait d'un premier pas vers une compensation ambisonique de la réverbération.

5. OPERATIONS MUSICALES DANS LE DOMAINE DES HARMONIQUES SPHERIQUES

Comme nous avons pu le voir dans la partie précédente, une représentation du champ sonore dans le domaine des harmoniques sphériques offre l'accès à un certain nombre de nouveaux paramètres variables dont le musicien pourra s'emparer à des fins expérimentales. Nous allons ici présenter deux exemples assez classiques relatifs à une utilisation détournée de l'ambisonie.

5.1. Contrôle de la résolution angulaire

Comme nous avons pu le noter précédemment, plus l'ordre du système ambisonique est élevé, plus le champ sonore est synthétisé avec fidélité. Nous avons cependant pu remarquer un attachement de certains compositeurs au "format B" (ordre 1) de par sa capacité à "envelopper" l'auditeur. Il s'agit en réalité d'un effet "low-fi" dû à une assez faible résolution angulaire du système. À des fins artistiques, il serait intéressant de pouvoir reproduire cet effet dynamiquement dans un cadre de travail ambisonique d'ordre supérieur.

La figure 5 illustre l'une des solutions possibles. Il s'agit de pondérer les différents signaux ambisoniques en fonction de leur ordre. En variant l'importance des différentes harmoniques, nous pouvons moduler la résolution angulaire de manière continue pour retrouver par exemple les caractéristiques d'un encodage au format B. Artistiquement, une métaphore optique pourrait être proposée en associant à une source sonore l'idée d'un faisceau dont nous pourrions modifier l'ouverture et la largeur à volonté.

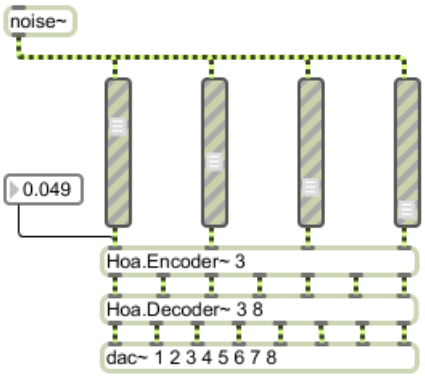


Figure 5. Pondération des composantes spatiales en fonction de leur ordre

5.2. Création d'un champ acoustique diffus

En acoustique, un champ diffus est un champ où l'énergie volumique moyenne a la même valeur en tous les points et où l'intensité acoustique est la même dans toutes les directions. La partie tardive de la réverbération d'une salle est un cas concret où nous pouvons noter la présence d'un champ diffus. Du point de vue de l'auditeur on note une sensation d'enveloppement, ce dernier ne peut en effet associer le champ sonore perçu à un nombre déterminé de sources sonores présentes dans l'espace.

Afin de reproduire un champ sonore diffus dans un cadre ambisonique, Jérôme Daniel propose d'insérer des signaux décorrélés (e.i. qui ont une covariance nulle) en entrée d'un décodeur ambisonique [4]. Il s'agit d'une proposition faite afin de modéliser les réflexions tardives d'une réverbération, nous pouvons cependant nous désolidariser de cette application et proposer la création d'un champ diffus à partir de tout signal monophonique. La problématique est alors de savoir comment obtenir plusieurs signaux décorrélés à partir d'un seul et unique signal.

Graham Wakefield suggère un exemple où cette décorrélation est effectuée via une banque de filtres passe-bande [5], chaque bande fréquentielle étant associée à un signal ambisonique convoyé au décodeur.

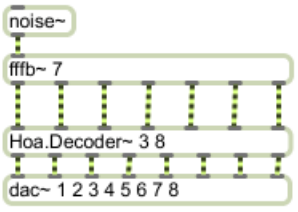


Figure 6. Décorrélation via une banque de filtres à destination d'un décodeur ambisonique

La figure 6 illustre la proposition faite par Wa-

kefield. D'autres solutions sont cependant possibles. Nous pourrions ainsi considérer l'utilisation d'un granulateur temps réel polyphonique répartissant aléatoirement chaque grain sur l'un des canaux du décodeur. Les méthodes possibles sont en réalité assez nombreuses et une évaluation artistique de ces dernières serait nécessaire afin de déterminer quelles sont les plus pertinentes dans un contexte musical.

6. CONCLUSION

La principale motivation de cet article a été de présenter Hoa.Lib, une nouvelle bibliothèque d'objets MAX/MSP dédiée à l'ambisonie d'ordre supérieur. Il s'agit d'un outil proposant certaines fonctionnalités assez avancées telles que la compensation du champ proche et la possibilité d'accéder à un ordre de décomposition non limité. Afin de détailler ce que ces fonctionnalités impliquent, une synthèse théorique des mécanismes liés à l'ambisonie a été proposée.

Nous avons par la suite présenté la bibliothèque avec davantage de détails en soulignant notamment son architecture modulaire. Cette dernière permettant à l'utilisateur d'optimiser les traitements en fonction de ses besoins. Elle laisse de plus l'accès à un grand nombre de paramètres susceptibles d'être utilisés à des fins artistiques expérimentales. La dernière partie de cet article a en effet présenté deux applications où l'usage détourné d'un moteur ambisonique permet d'aboutir à de nouvelles formes artistiquement pertinentes. C'est dans cette direction que nous souhaitons d'ailleurs continuer nos recherches, via une évaluation artistique des différentes possibilités offertes par une représentation du champ sonore dans le domaine des harmoniques sphériques. Cette évaluation s'effectuera sur le terrain, en interaction avec les différents compositeurs du CICM. Hoa.Lib peut ainsi être vue comme le cadre de travail nécessaire à nos prochains travaux.

7. REFERENCES

[1] Sedes, A. et al. *Espaces sonores : Actes de recherches*. Transatlantiques (éditions musicales), Paris, 2003.

[2] Sedes, A. Courribet, B. et Thiébaut, J. B. "Ego-sound, and egocentric, interactive and real-time approach of sound space", *Proceedings of the 6th International Conference of Digital Audio Effect*, London, UK, 2003.

[3] Gerzon, M. "Ambisonics in Multichannel Broadcasting and Video", *Journal of the Audio Engineering Society*, Plymouth, USA, 1985.

[4] Daniel, J. "Spatial Sound Encoding Including Near Field Effect : Introducing Distance Coding Filters and a Viable, New Ambisonic Format", *AES 23rd International Conference*, Copenhague, Danemark, 2003.

[5] Wakefield, G. "Third-Order Ambisonic Extensions for Max/MSP with Musical Applications", *International Computer Music Conference*, University California Santa Barbara, USA, 2006.

[6] William, E. G. *Fourier Acoustics : Sound Radiation and Nearfield Acoustical Holography*. Academic Press, London, 1999.

[7] Gumerov, N. A. Duraiswami, R. *Fast Multipole Methods For The Helmoltz Equation In Three Dimensions*. Elsevier, University of Maryland, 2004.

[8] Colafrancesco, J. "Caractérisation du champ sonore dans les salles, applications à la mesure et au transcodage de réponses impulsionnelles multicanaux" *Master Thesis*. IRCAM, Centre George Pompidou, 2011.

[9] Zotter, F. "Analysis and synthesis of sound-radiation with spherical arrays" *Phd Thesis*. University of Music and Performing Arts, Austria, 1999.

[10] Golub, G. H. Kahan, W. "Calculating the singular values and pseudo-inverse of a matrix", *Society for Industrial and Applied Mathematics : Journal on Numerical Analysis*, 2 ser. B p. 205-224, 1965.

UN MODÈLE INTERMÉDIAIRE DYNAMIQUE GÉNÉRATIF BASÉ SUR L'AUTOMATE CELLULAIRE DU « JEU DE LA VIE »

Vincent Goudard
vincent@mazirkat.org

Boris Doval
boris.doval@upmc.fr

Hugues Genevois
genevois@lam.jussieu.fr

LAM, Institut Jean Le Rond d'Alembert

RÉSUMÉ

Le découplage entre le geste de l'instrumentiste et le son produit par son instrument dans les musiques électroniques a entrainé une reconsidération totale de la notion d'instrument [5] et de nouvelles pratiques pour la facture d'instruments numériques [1].

Pour concevoir des instruments numériques riches, le facteur est souvent amené à réaliser des connexions complexes et des réglages très fins, afin de retrouver à la fois un son vivant et une synesthésie entre le geste et le résultat acoustique qui permet *in fine* de pouvoir anticiper ou "chanter" ce que l'on joue.

Il est possible d'obtenir un couplage riche et intuitif en procédant empiriquement au câblage et aux courbes de transfert entre toutes les variables d'entrée et de sortie, mais cette opération fastidieuse se fait souvent au détriment de la modularité informatique, qui permettrait de changer d'interface ou de synthèse en un simple clic.

Les auteurs, qui ont déjà présenté le concept de "Modèles Intermédiaires Dynamiques" (MID) pour répondre à ce problème, présentent dans cet article leurs derniers développements concernant le contrôle de macro-formes par des processus génératifs. L'utilisation d'un automate cellulaire connu sous le nom de "Jeu de la Vie" est plus particulièrement présentée ici.

1. INTRODUCTION

1.1. Le projet OrJo

Dans le cadre du programme OrJo[1], qui vise à réaliser des instruments numériques audio-graphiques destinés au jeu collectif, et pouvant s'intégrer au logiciel Méta-Mallette [9], les auteurs s'attachent à développer une famille d'instruments modulaires basés sur le concept de modèle intermédiaire dynamique (MID).

Un modèle intermédiaire est un modèle instrumental qui contrôle des synthèses audio, graphiques ou encore électro-mécaniques à partir des actions de l'instrumentiste sur un contrôleur électronique ou une interface gestuelle (eg. clavier MIDI, joystick, tablette graphique, etc.).

Il s'insère dans la chaîne de mapping qui relie les valeurs issues des capteurs gestuels aux paramètres de synthèse, tout en se différenciant d'un simple "conditionnement de données" par sa nature complexe, dynamique et non-linéaire.

Ce modèle est donc (relativement) indépendant des types de synthèse qu'il contrôle, ainsi que de leur nombre, et nous émettons l'hypothèse qu'il est possible d'apprendre son comportement (relativement) indépendamment de l'interface en amont et de la synthèse en aval, en se construisant une image mentale de la cinématique du modèle manipulé.

Une telle architecture a aussi pour objectif de favoriser des apprentissages implicites (comme c'est le cas pour la plupart des apprentissages supposant une activité psycho-motrice contrôlée) alors que les techniques "classiques" de mappings statiques supposent le plus souvent un apprentissage explicite.

A travers ce concept de MID, nous avons l'ambition d'adresser des "gestes musicaux" recouvrant aussi bien ceux de la performance, que ceux de la composition, de l'accordage, de la lutherie et de l'écoute, dans un même environnement. Ces différents types de gestes sont souvent intimement mêlés dans le processus de création musicale, et il nous parait indispensable de pouvoir composer de toute leur richesse sans introduire de clivage entre eux. Jouer de l'écoute, composer du geste, accorder les modes de jeu… sont des propositions musicales qui ne sont pas de simples figures de style littéraires. En effet, les processus musicaux que l'on cherche à contrôler se déploient sur une très large échelle temporelle, allant de la micro-structure du son jusqu'au pilotage de motifs musicaux, c'est-à-dire la génération et la transformation de formes musicales sur plusieurs secondes, voire plusieurs minutes.

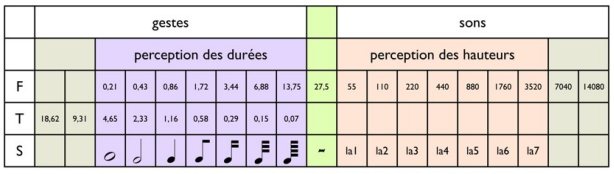


Figure 1. Représentation d'une partie du "continuum" temporel F : fréquences en Hz ; T : durées en secondes ; S : symboles musicaux correspondants

[1]Le programme OrJo réunit l'association Puce Muse, le LAM (Institut Jean le Rond d'Alembert), le LIMSI et la société 3Dlized. Il fait l'objet d'un soutien du FEDER et du Conseil Régional d'Ile-de-France.

Les modèles intermédiaires utilisés doivent donc pouvoir s'adapter à ces différentes échelles, ainsi qu'aux gestes mis en œuvre par le musicien. Pour cela, nous avons décidé d'explorer différentes familles de modèles dynamiques : physiques (simulant le comportement de structures virtuelles susceptibles d'être décrites par des lois physiques), topologiques (navigation dans des espaces virtuels), génétiques (basés sur des règles gérant le comportement des objets manipulés : naissance, croissance, évolution, disparition), etc.

Dans un premier temps, nous avons développé un certain nombre de modèles intermédiaires destinés au contrôle de la synthèse à une échelle temporelle correspondant à la modulation du timbre ou micro-rythmique. Les travaux réalisés dans le cadre de cette première phase ont fait l'objet de présentations à l'occasion de congrès récents (JIM 2011, SMC 2011, DAFx) [7], et nous n'y reviendrons pas dans le présent article.

Dans un deuxième temps, nous avons développé des modèles intermédiaires agissant à une échelle temporelle plus grande, en nous intéressant à des algorithmes de croissance génétique. Notamment, nous avons tenté d'utiliser des automates cellulaires tel que le "jeu de la vie" de John Horton Conway [6].

2. MID BASÉ SUR UN AUTOMATE CELLULAIRE : JEU DE LA VIE

2.1. Caractéristiques du "jeu de la vie"

Le "jeu de la vie" est un automate cellulaire très simple et très fertile. Une très vaste littérature décrit son histoire et ses caractéristiques, ainsi que son utilisation en informatique musicale [4], nous laisserons le lecteur s'y référer pour plus de détails, en rappelant uniquement les deux règles sur lesquels il se base :

- Une cellule morte possédant exactement trois voisines vivantes devient vivante.
- Une cellule vivante possédant deux ou trois voisines vivantes le reste, sinon elle meurt.

Si l'univers du "jeu de la vie" est théoriquement infini, il est courant d'utiliser un univers fini, quoique non borné, en re-bouclant un plan sur lui même pour obtenir un tore. Cette technique permet à la fois de s'affranchir du problème des limites, ainsi que de disposer des valeurs des cellules sous forme d'une matrice de dimensions connues.

Si, d'un point de vue théorique, l'utilisation d'automates cellulaires n'est pas sans difficultés pour ce qui nous concerne ("bizarreries" thermodynamiques : gestion hasardeuse de l'entropie, et topologiques : incidence des limites et re-bouclage du plan), elle n'en reste pas moins extrêmement intéressante pour la gestion de générations/transformations morphologiques à différentes échelles spatiales et temporelles.

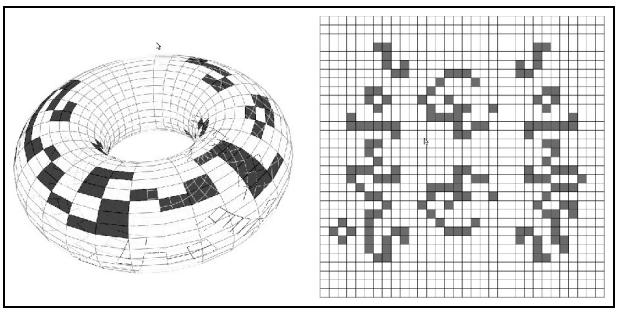


Figure 2. L'oscillateur "diuresis" représenté dans sur une surface bouclée (tore) et déroulée en plan.

Le Jeu de la Vie est un processus essentiellement discret dans le domaine spatial et temporel, mais son mode de prolifération par voisinage lui confère une continuité lorsque l'on utilise un canevas très large.

2.2. Modes d'intervention

Le "jeu de la vie" qui se déploie dans un espace 2D a l'avantage d'être bien adapté aux surfaces de contrôle telles que la tablette graphique ou les écrans tactiles multi-points dont l'usage se répand depuis quelques années. Dans une optique d'utilisation instrumentale d'un tel modèle, l'ergonomie de l'édition des données prend toutefois une importance cruciale. Pour la performance "live", un certain nombre de fonctionnalités nous ont paru utiles :

- éditer le canevas en de multiples endroits à la fois
- sélectionner des zones particulières pour les altérer sans toucher au reste du canevas
- pré-visualiser son geste de sélection, et ne le déclencher qu'au moment opportun.
- revenir à des états mémorisés
- introduire des boucles
- insérer directement des figures aux comportements identifiés

Nous avons accordé une importance toute particulière à ce dernier point en considérant l'importante thésaurisation réalisée par les chercheurs dans le domaine des automates cellulaires ainsi que les enthousiastes du "jeu de la vie".

2.3. L'utilisation du lexique de figures

En effet, les automates cellulaires en général et le "jeu de la vie" en particulier ont été largement étudiés depuis les années 1970, et restent un domaine de recherche et de développement très actif[2]. Il existe à ce jour un "bestiaire" de plus de 700 figures identifiées dans ce paradigme. Or, malgré un nombre important de recherches sir la génération de musique par automate cellulaire (cf. bibliographie dans [3]), l'importance de

[2] Cf. le développement de Golly (http://golly.sourceforge.net), et les sites http://conwaylife.com ou encore http://pentadecathlon.com

cet inventaire semble avoir été relativement ignorée. Cet inventaire est pourtant l'outil de base d'une programmation extrêmement complexe avec ce langage Turing-complet et permet de réaliser des constructions extrêmement complexes.

Cette classification présente l'intérêt pédagogique d'être mémorisable, grâce aux noms originaux et pittoresques qui ont été attribués aux différentes figures ("le glisseur", "le bateau", "le mathématicien", "le moule", etc...). Également, un certain nombre de propriétés et de grandeurs quantitatives des figures ont été nommées : boîte englobante, période, vitesse, nombre de cellule, chaleur, type de figure, volatilité.

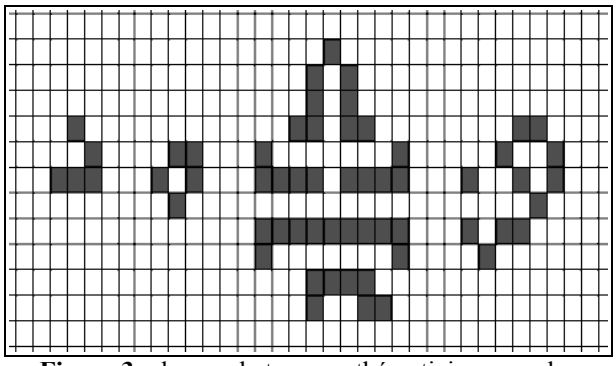


Figure 3. planeur, bateau, mathématicien, moule.

2.3.1. Le cas des oscillateurs

Les oscillateurs sont des figures ayant la caractéristique de retourner à son état d'origine, dans la même orientation et à la même position, au bout d'un nombre fini de générations., que l'on nomme "période" de l'oscillateur.

En "déroulant" un oscillateur temporellement, on obtient une sorte de partition qui aide à saisir d'un coup d'œil des caractéristiques d'un oscillateur particulier telles que :

- les rythmes internes de chaque cellule par rapport au rythme global
- l'indépendance relative des rythmes de chaque cellule
- les symétries de l'oscillateur
- la volatilité de l'oscillateur (c'est à dire la proportion de cellule qui oscille effectivement par rapport aux cellules restant fixe)
- la chaleur de l'oscillateur (c'est à dire la quantité de cellules qui change à chaque génération)

Dans une perspective d'utilisation pour le jeu temps-réel, le déploiement temporel d'un oscillateur permet également de le sélectionner à un moment particulier de son oscillation (sa phase) pour l'insérer dans une grille de "jeu de la vie". Ce qui revient, formulé en termes plus musicaux, à pouvoir insérer une boucle rythmique en commençant par une anacrouse.

2.3.2. Le cas des "vaisseaux"

Il existe un autre type de figure particulière, qui est une forme d'oscillateur mais dont la révolution complète le fait revenir à sa forme initiale, mais de manière décalée par rapport à sa position d'origine. La conséquence est que cet oscillateurs se déplace, et on lui a donné pour cette raison le nom de "vaisseau" ou "spaceship" en anglais. Ce type de figure possède également une période qui représente le nombre de générations au bout duquel on le retrouve avec une forme identique. Il possède aussi une vitesse, exprimée dans un rapport c/N ou c représente la vitesse limite de déplacement, c'est à dire 1 cellule par génération, ainsi qu'une direction de déplacement : orthogonale ou diagonale.

Chaque vaisseau possède sa propre façon de se mouvoir. Il en résulte une fonction iso-rythmique caractéristique, motif composé de progressions et de reculs propre au vaisseau, qui sur un canevas polarisé selon une gamme de hauteurs se traduit en une forme arpégées.

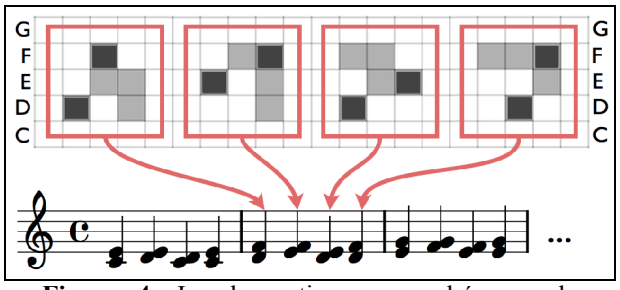


Figure 4. Iso-chromatisme engendrés par la progression du *Planeur*. Les notes correspondent à un espace polarisé verticalement en ne prenant en compte que la valeur de la ligne. Cette représentation ne représente pas nécessairement des notes de la gamme, mais des événements associés à un index de ligne.

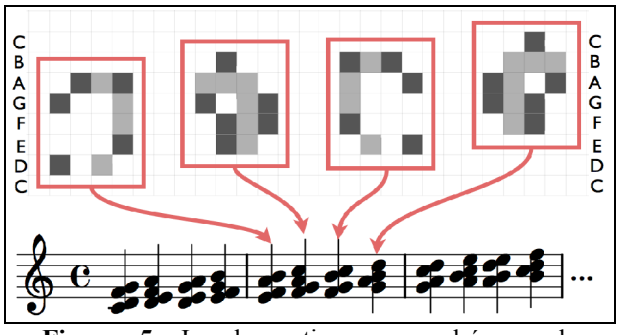


Figure 5. Iso-chromatisme engendré par la progression du "*Light Weight Space Ship*" (LWSS) représenté selon la même modalité que la figure précédente.

Une autre fonction des *vaisseaux* est d'aller modifier le canevas à un autre endroit, en réagissant au contact d'autres cellules.

2.3.3. Autres types de figures

Il existe des figures instables dont les comportement est également intéressant dans une perspective musicale. Les *mathusalems* sont des figures explosives qui à partir d'un petit nombre de cellules initiales s'étendent rapidement à des formes plus complexes, avant de se stabiliser voire de disparaître complètement.

Les *canons* sont des oscillateurs produisant des *vaisseaux*, les *puffeurs* sont des *vaisseaux* qui laissent des débris derrières leur passage et les *breeders* sont des *vaisseaux* générant des *canons*. Les *mèches* sont des figures linéaires qui se "consument" en un nombre paramétrable de générations et permettent d'introduire de la sorte des retards dans le chaînage de différentes figures. Les *jardins d'Eden* sont des figures ne possédant aucun prédécesseur.

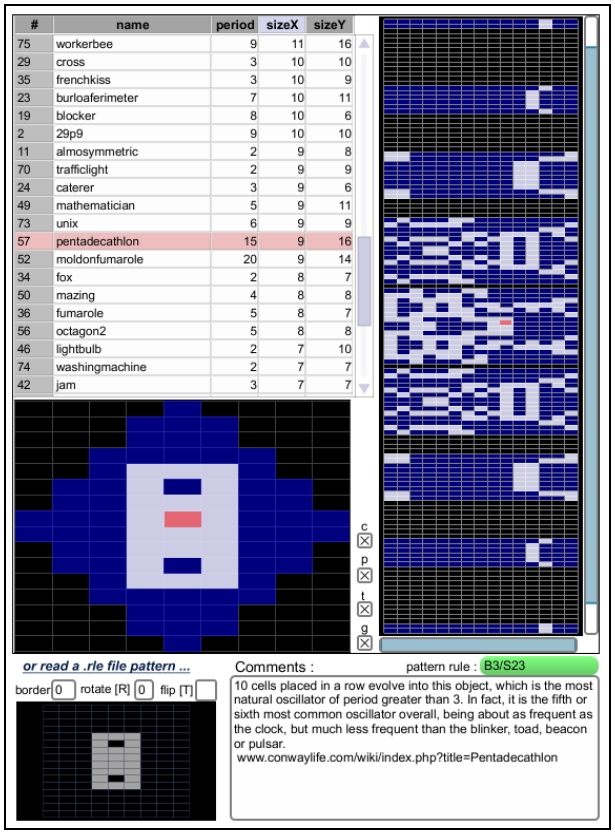
Enfin, il existe un certains nombre de figures stables ou non, qui dans le cas d'une programmation qui utilise les différents types de figures comme des fonctions interconnectées, se mettent à remplir un rôle particulier ("*tagalong*", "*sidecar*", "*fuse*").

2.3.4. Banque de figures

Le catalogue des figures a l'avantage d'exister, mais l'inconvénient de ne pas avoir spécialement été conçu dans une optique musicale. Il nous a donc paru utile de proposer ce catalogue dans un format plus adapté à l'utilisation des figures pour la composition temps-réel.

Nous avons ainsi développé un lecteur de fichier au format RLE, format dans lequel sont codées la plupart des figures du jeu de la vie. Ce module permet d'accéder à un catalogue de plus de 700 figures recensées et disponible sur internet[3] directement dans l'environnement Max/MSP.

Des outils d'analyse permettent ensuite de déduire la période et la taille des figures, ainsi que de les développer temporellement. Enfin, des outils de transformation spatiale permettent d'orienter la figure à sa convenance avant de l'insérer dans le canevas de calcul.



#	name	period	sizeX	sizeY
75	workerbee	9	11	16
29	cross	3	10	10
35	frenchkiss	3	10	9
23	burloaferimeter	7	10	11
19	blocker	8	10	6
2	29p9	9	10	10
11	almosymmetric	2	9	8
70	trafficlight	2	9	9
24	caterer	3	9	6
49	mathematician	5	9	11
73	unix	6	9	9
57	pentadecathlon	15	9	16
52	moldonfumarole	20	9	14
34	fox	2	8	7
50	mazing	4	8	8
36	fumarole	5	8	7
56	octagon2	5	8	8
46	lightbulb	2	7	10
74	washingmachine	2	7	7
42	jam	3	7	7

Figure 6. En haut à gauche, le tableau permettant de trier les figures selon leurs caractéristiques. En dessous la figure, et à droite sa "partition". En blanc : la figure à un certain moment de sa phase, en bleu : le terrain occupé par la figure durant sa révolution, en rouge : un curseur qui permet de se déplacer de manière synchrone dans les 2 représentations spatiale et temporelle.

3. INTERACTION AVEC LA SYNTHESE

3.1. Plusieurs stratégies possibles

Le résultat produit par le jeu de la vie est essentiellement une matrice. De nombreuses manières de d'interpréter le canevas de jeu de la vie en vue de le sonifier ont été expérimentées [2], comme le calcul de valeurs statistiques, la lecture de la matrice comme une table d'onde ou de correspondance (look-up table), ou encore l'utilisation de coordonnées polaires [8].

L'utilisation du MID basé sur le jeu de la vie au sein d'un environnement de programmation modulaire comme la Méta-Mallette nous laisse la possibilité d'expérimenter avec chacune de ces méthodes, d'en inventer d'autre, ou encore d'insérer le jeu de la vie à l'intérieur d'un réseau plus complexe de modèles intermédiaires.

Dans l'optique de contrôler des paramètres de macro-formes comme la génération de phrases musicales, nous avons délaissé les techniques basées sur une interprétation de la matrice comme un signal audio et

[3] Une importante collection est notamment disponible sur le site http://conwaylife.com/wiki/

principalement expérimenté une logique de déclenchement d'événements.

3.2. Grille de déclencheurs

Pour sonifier le modèle de jeu de la vie, nous avons développé un modèle intermédiaire topologique en aval consistant en une grille de déclencheur.

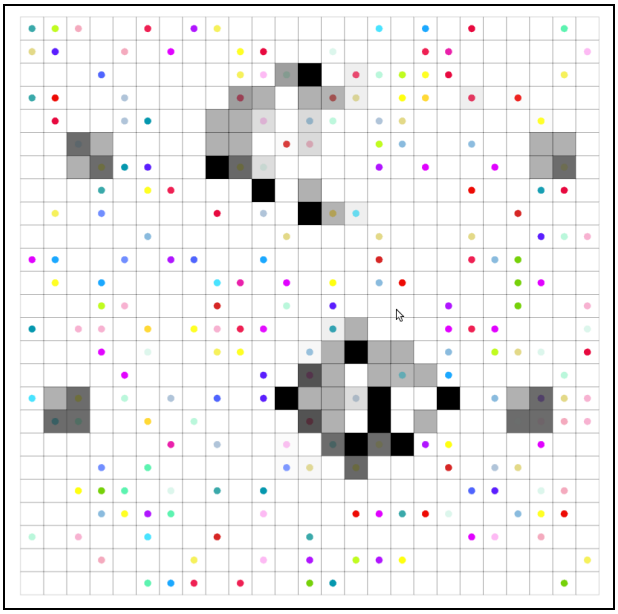


Figure 7. Deux "Queen Bee Shuttles" en action déphasée sur un canevas criblé de déclencheurs. La surbrillance indique les déclenchements.

Ce modèle permet de positionner des déclencheurs sur une matrice qui agit comme un crible en opérant une intersection avec la matrice à traiter. Dans notre cas, les déclencheurs seront activés lorsqu'une cellule du jeu de la vie change d'état.

Les déclencheurs peuvent par exemple engendrer des notes MIDI, mais plus généralement n'importe quel type de message, comme un changement de tempo, une transposition, et même la modification des matrices de déclencheurs et du canevas de calcul.

Cette flexibilité et sa récursivité ouvre un vaste espace de possibles dont l'appréhension varie entre une construction volontariste, réfléchie et l'observation d'une émergence semi-chaotique.

Pour donner un exemple simple d'utilisation possible de ce genre de logique, on peut imaginer utiliser un oscillateur comme le "pentadecathlon" (de période 15) pour jouer une mélodie de notes, et assigner à une cellule particulière un événement qui transpose alternativement une tierce plus haut, puis une tierce plus bas.

En utilisant une combinaison de cribles auto-récursifs ainsi qu'un vocabulaire de figures dont on connaît les périodes, il est possible d'arriver à un contrôle temps réel de la synthèse musicale par des paramètres de haut niveau. Sans pour autant contrôler chaque note, chaque rythme, il est possible de composer des topologies à plusieurs gammes, dont chacune intervient selon un degré de probabilité, ou encore de privilégier certains rapports rythmiques, d'agir sur des événement très localisés, ou au contraire d'influence globale.

4. PERSPECTIVES

L'utilisation du jeu de la vie ouvre un espace immense dans lequel les choix devront probablement s'affiner de manière empirique. Beaucoup de choses restent à inventer sur l'ergonomie d'une manipulation temps-réel de ce modèle, et sur les connexions possibles avec la synthèse musicale en aval. Dans cette perspective, on peut espérer que l'intégration des Modèles Intermédiaires Dynamiques dans l'instrumentarium de la Méta-Mallette et son utilisation pratique par un public plus large fassent émerger des propositions intéressantes de ce vaste espace de possibles. En particulier, on peut imaginer que des topologies de grilles de déclencheurs plus efficaces que d'autres viennent émerger.

Les performances limitées de l'implémentation du jeu de la vie dans Max/MSP nous privent également de l'utilisation de figures hyper-complexes, telles que celles qui ont pu être développées dans des logiciels plus optimisés comme Golly. Des phénomènes tout à fait singuliers semblent émerger dans des canevas de plusieurs millions de cellules, et il ne semblent pas impossible d'appréhender ces figures hyper-complexes, car certaines sont purement construites par assemblage d'éléments simples.

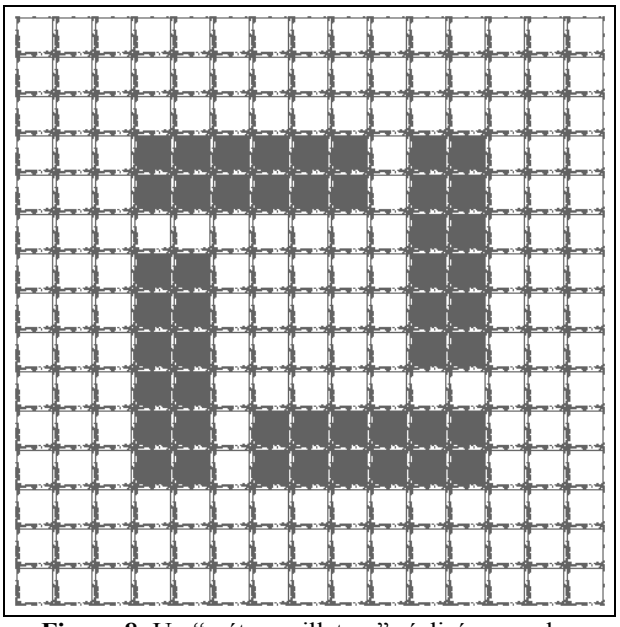


Figure 8. Un "méta-oscillateur" réalisé avec plus de 7 millions de cellules dans Golly. Chaque "méta-cellule" active est composée d'environ 65000 cellules.

5. RÉFÉRENCES

[1] Battier, M. *« L'approche gestuelle dans l'histoire de la lutherie électronique. Étude de cas : le theremin »,* in Les nouveaux gestes de la musique, H. Genevois et R. de Vivo (éds). Éditions Parenthèses, 139-156, 1999.

[2] Beyls, P, « Cellular Automata Mapping Procedures », Proceedings of the ICMC, 2004

[3] Burraston, D., Edmonds, EA, Livingstone, D. et Miranda, E. "Cellular Automata in MIDI based Computer Music" Proceedings of the International Computer Music Conference, pp. 71-78, 2004.

[4] Burraston, D., et E. Edmonds, "Cellular Automata in Generative Electronic Music and Sonic Art: A Historical and Technical Review." Digital Creativity 16(3):165-185, 2005

[5] Cance, C., Genevois, H., Dubois, D. « What is instrumentality in new digital musical devices? A contribution from cognitive linguistics and psychology », Proceedings of CIM09, to be published « La musique et ses instruments » (2012).

[6] Gardner, M., « Mathematical Games. The fantastic combinations of John Conway's new solitaire game "life" », Scientific American no 223 (Octobre 1970), p. 120-123.

[7] Goudard, V., Genevois, H., Ghomi, E., Doval, B., « Dynamic Intermediate Models for Audiographic Synthesis », SMC 2011, Padova, Italie, 2011

[8] Kirke, A., Miranda, E. R., Capturing the aesthetic: Radial mappings for cellular automata music. J. ITC Sangeet Res. Acad. 21, 15--23. 2007

[9] de Laubier, S., Goudard, V., « Puce Muse - La Méta-Mallette », Proceedings of Journées d'Informatique Musicale (JIM 2007), Lyon, 2007

LA COMPOSITION ÉLECTROACOUSTIQUE POUR INTERFACE INVENTÉE

Martin Marier
Université de Montréal
Centre interdisciplinaire de recherche en musique, médias et technologie (CIRMMT)

RÉSUMÉ

L'auteur explique son approche de la composition et de l'interprétation de la musique électroacoustique. Il cherche à retrouver le côté ludique du jeu instrumental et à améliorer l'interaction avec le public en utilisant une nouvelle interface musicale pour interpréter ses ses oeuvres. L'interface qu'il a conçue s'appelle l'éponge et est, en quelque sorte, un coussin muni de capteurs qui en détectent les déformations. Les stratégies de *mapping* qui ont été développée sont décrites. Les difficultés et enjeux de la composition électroacoustique pour nouvelles interfaces sont abordés et discutés.

1. INTRODUCTION

Dans les années '40, les nouvelles technologies permirent à Pierre Schaeffer de découvrir la musique concrète [13], celle qu'on appela plus tard musique acousmatique ou musique électroacoustique. Il était désormais possible de composer des musiques qui n'agençaient plus uniquement les sons instrumentaux, mais aussi tous les sons susceptibles d'être enregistrés ou synthétisés.

Cet art des sons fixés a donné naissance à l'art de la diffusion qui est toujours pratiqué aujourd'hui. Dans la salle de concert, où une multitude de haut-parleurs sont disposés, l'interprète spatialise le son, mais n'est responsable ni de sa génération, ni du déroulement temporel. C'est la fixation du son sur support qui, à l'origine, rendait tous les sons accessibles au musical. Mais de nos jours, la technologie permet une manipulation en direct non seulement de l'espace et de la dynamique, mais aussi d'une multitude d'autres paramètres du son. Il serait donc possible d'interpréter réellement la musique électroacoustique ; de la jouer de la même façon qu'une pièce instrumentale.

Mais l'accès à cette technologie ne règle pas tous les problèmes. Il faut aussi disposer d'une interface gestuelle qui permettra un contrôle expressif de tous ces paramètres, et c'est là le coeur de ma recherche. À cet effet, j'ai conçu une interface musicale que j'ai appelée l'éponge.

Afin de mieux comprendre ce qu'est l'expressivité et comment il est possible de l'obtenir, j'ai développé les notions de « contrôle musical » et de « contrôle paramétrique » que je décrirai plus précisément à la section 2. Je présenterai ensuite l'interface musicale que j'ai conçue et fabriquée : l'éponge. Après ce survol technique, j'élaborerai les différents enjeux et difficultés associés à cette approche : la pérennité des oeuvres, la multidisciplinarité et surtout, les stratégies qui permettent d'établir un rapport geste-son clair. Je parlerai ensuite des compositions qui ont été écrites pour l'éponge et, pour finir, les avenues possibles pour le futur seront présentées.

2. CONTRÔLE PARAMÉTRIQUE VS CONTRÔLE MUSICAL

Au cours de mes recherches, j'ai développé les notions de « contrôles musical » et de « contrôle paramétrique ». Je les définies comme étant les deux extrémités d'un continuum sur lequel pourrait se trouver une grande variété de types de contrôle.

Le contrôle musical permet le transfert direct du caractère du geste vers le caractère du son, mais ne permet pas de contrôler précisément la valeur d'un paramètre. Sur une guitare acoustique, par exemple, il est relativement facile de rejouer une même phrase musicale en lui donnant chaque fois un caractère différent (agressif, bondissant, sombre, brillant, etc.). Par contre, il est impossible, et ce même pour un interprète aguerri, de jouer deux notes rigoureusement identiques, mais dont l'une serait précisément quatre décibels plus forte que l'autre.

Le contrôle paramétrique, quant à lui, permet ce degré de précision, mais ne permet pas le transfert du caractère du geste. Si, par exemple, on utilise le clavier alphanumérique d'un ordinateur pour entrer la fréquence exacte d'un oscillateur, on est bien en présence d'un contrôle paramétrique. À ce niveau extrême, le geste est complètement divorcé du son.

Une fois ces deux extrêmes établis, il est relativement aisé de classer les différents contrôles dans le continuum.

Le potentiomètre d'une table de mixage offre un contrôle légèrement plus musical que le clavier d'ordinateur puisqu'un aspect du geste est relié à un aspect du son. Cependant, l'énergie du geste n'est pas transférée à l'énergie du son. Si c'était la vitesse de déplacement du potentiomètre qui affectait l'amplitude, on se rapprocherait encore du contrôle musical.

La référence en ce qui concerne le contrôle musical, c'est le corps sonore. Tout objet qu'on peut manipuler physiquement et qui produit un son acoustique offre forcément un contrôle musical. Paradoxalement, ce qui différencie les instruments acoustiques de tous les autres corps sonores, c'est qu'ils permettent le contrôle précis de cer-

tains paramètres musicaux, comme par exemple, celui de la hauteur.

Le défi des luthiers traditionnels, c'est de fabriquer des instruments acoustiques qui offrent un contrôle paramétrique, alors que la difficulté en lutherie numérique, c'est de fabriquer des instruments qui offrent un contrôle musical.

2.1. Les problèmes du contrôle paramétrique

Le contrôle paramétrique est très utile et souhaitable lors d'un certain type de travail en studio comme, par exemple, les étapes de mixage et de mastering qui ont bien besoin de ce genre de précision. Par contre, ce type de contrôle est mal adapté à l'interprétation ou à l'improvisation. Il comporte trois principales faiblesses.

2.1.1. Le côté ludique est atrophié

Comme il n'y a pas de transfert direct du caractère du geste au caractère du son, le contrôle paramétrique requiert une étape d'analyse avant l'intervention. Il est évident que jouer d'un instrument acoustique requiert aussi un travail intellectuel, mais insuffler un caractère au son ne se fait qu'en un seul geste. Inversement, insuffler un caractère en utilisant un séquenceur audio requiert la coordination et l'ajustement précis d'une multitude de courbes d'automatisation. Cette tâche laborieuse, recherchée avec soin, est longue à mettre en oeuvre et empêche donc l'approche impromptue du jeu musical traditionnel.

2.1.2. Obtenir des caractères musicaux est très difficiles

Si nos outils n'offrent que des contrôles proches du contrôle paramétrique, transformer un caractère *agressif* en un caractère *scintillant*, par exemple, sera extrêmement difficile. Si on veut arriver à ce résultat en utilisant un séquenceur audio traditionnel, il faudra encore une fois ajuster et coordonner une multitude de courbes d'automatisation.

Malheureusement, en musique électroacoustique, le contrôle musical pur ne se trouve qu'à l'étape de la prise de son, lors de la manipulation des corps sonores. La grande majorité des outils logiciels disponibles à ce jour n'offrent que des contrôles plus proches du contrôle paramétrique.

2.1.3. L'interaction avec le public est minimale

Dans le cadre d'une expérience, Wanderley et collab. [15] ont demandé à des participants de regarder, d'écouter ou de regarder *et* écouter la performance d'un clarinettiste. La perception que les participants avaient de la tension musicale et des phrasés musicaux était enregistrée en temps réel. En outre, Wanderley et colllab. [15] ont observé que les gestes de l'interprète ne faisaient pas que suivre les phrases musicales, ils les prolongeaient dans le silence. Les résultats de cette expérience ont montré que les participants qui voyaient l'interprète comprenaient mieux les messages musicaux qui étaient véhiculés. Les informations sur la tension et les phrasés seraient donc communiquées autant par le geste que par le son.

À cela, me basant sur mon expérience personnelle, j'ajouterais que la seule présence d'un interprète ne suffit pas. Quand le contrôle qu'exerce le performeur est trop loin du contrôle musical, ses gestes sont divorcés du son et ne contribuent pas à communiquer le message. Lors d'une performance de *live coding* [1], par exemple, les gestes et les expressions faciales du codeur sont complètement détachés du son produit.

3. L'ÉPONGE : VERS UN CONTRÔLE MUSICAL DE LA MUSIQUE ÉLECTROACOUSTIQUE

Figure 1. L'éponge.

L'éponge est une interface musicale que j'ai développée dans le but de permettre l'interprétation devant public de pièces électroacoustiques. Elle ressemble en tout point à un coussin rayé (voir la figure 1), sauf que des capteurs installés à l'intérieur détectent les chocs qu'elle subit, son inclinaison, ainsi que ses déformations (compression, torsion, pliage). Afin d'exploiter le potentiel qu'offre le médium électroacoustique, j'ai développé une interface dont la forme et l'aspect ne rappellent aucun instrument acoustique. Une interface calquée sur un instrument acoustique existant aurait permi d'exploiter l'expertise d'un instrumentiste aguerri [2]. Par contre elle aurait aussi conditionné le travail de création en imposant son mode de jeu, ce qui aurait considérablement biaisé les choix esthétiques.

En résumé, l'éponge a été conçue pour permettre d'interpréter la musique électroacoustique en permettant un contrôle musical.

3.1. Construction et capteurs

L'éponge comporte 11 capteurs analogiques : sept interrupteurs momentanés, deux capteurs de pression (FSR)

1 . Le *live coding* est une pratique qui consiste à utiliser un langage de programmation dédié à la musique pour performer devant un public. Traditionnellement, le code du performeur est projeté sur grand écran pour que le public puisse voir et apprécier son travail.

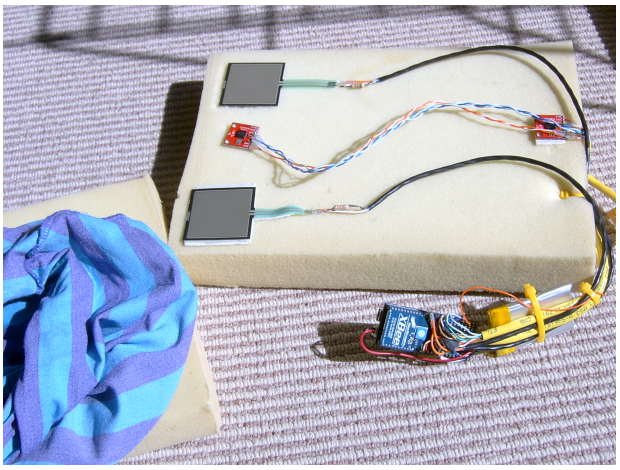


Figure 2. L'éponge nue. On peut voir les deux capteurs de pression (carrés gris), les accéléromètres (composantes rouges), l'interface Arduino Fio et le module XBee. Les sept boutons ne sont pas visibles sur cette photo ; ils sont sous l'éponge, du côté droit.

et deux accéléromètres 3D. Comme les deux accéléromètres sont sensibles aux trois dimensions, l'éponge fournit un total de 15 signaux.

Les capteurs de pression permettent de détecter la pression appliquée sur l'éponge (les carrés gris visibles sur la figure 2). Les accéléromètres (les composantes rouges visibles sur la figure 2) captent une multitude de déformations et mouvements. Ils servent d'abord à détecter l'inclinaison globale de l'éponge sur les trois axes, soit le roulis, le tangage et le lacet (illustrés par la figure 3). [2]

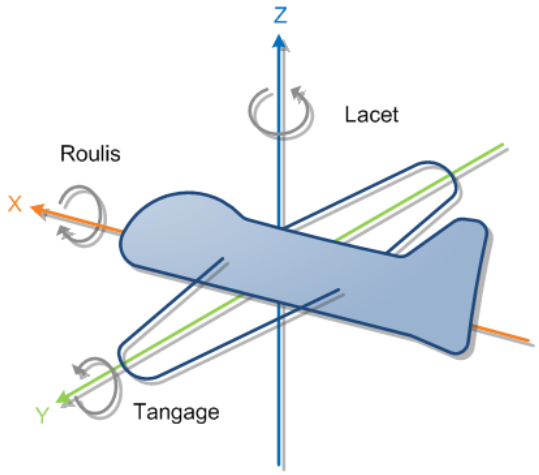


Figure 3. Les notions de roulis, tangage et lacet sont empruntées au domaine de l'aéronautique.

Ensuite, en calculant les différences d'inclinaison entre les deux accéléromètres, il est possible de déduire la torsion et le pliage. Les accéléromètres sont aussi utilisés pour détecter les chocs et les vibrations.

Les six axes des accéléromètres ainsi que les deux capteurs de pression sont connectés aux huit entrées analogiques d'une interface *Arduino Fio*. Les sept boutons sont connectés à sept entrées numériques de la même interface.

Chaque signal analogique est quantifié à 10 bits approximativement 300 fois par secondes et est acheminé vers l'ordinateur via une interface *XBee* (sans fil). Si les pilotes adéquats sont installés, l'éponge apparaît au système d'exploitation comme un périphérique sériel standard (`/dev/ttyUSB0`, par exemple), lui permettant de fonctionner sur les trois systèmes d'exploitation principaux, soit Linux, Mac OSX et Windows.

3.2. Le mapping

3.2.1. One-to-one *ou* many-to-many *?*

Le *mapping* consiste à établir une correspondance entre les paramètres du geste (les signaux issus des capteurs) et les paramètres des algorithmes de traitement ou de génération du son. C'est donc une étape incontournable de tout travail avec une interface musicale. Lors de la conception d'un *mapping*, on peut choisir l'une de ces quatre approches [5] :

one-to-one Un signal de contrôle n'affecte qu'un seul paramètre.

one-to-many Un signal de contrôle affecte plusieurs paramètres.

many-to-one Plusieurs signaux de contrôle n'affectent qu'un seul paramètre.

many-to-many Une combinaison des options précédentes : plusieurs signaux de contrôle affectent une multitude de paramètres.

Les *mappings* implémentés dans les outils électracoustiques traditionnels entrent habituellement dans la première catégorie. Sur une table de mixage, par exemple, le potentiomètre principal ne fait varier qu'un seul paramètre : l'amplitude.

Un bon exemple de *mapping one-to-many* est le potentiomètre d'interpolation de préréglage qu'on trouve dans les plugiciels de la série GRM Tools (figure 4). En actionnant ce seul potentiomètre, tous les autres paramètres du traitement sont affectés.

Toutes les parties logicielles de l'éponge sont des outils qui servent de près ou de loin à concevoir des *mappings* qui contribent à établir un contrôle musical.

3.3. Logiciel

La partie logicielle de l'éponge consiste en une bibliothèque implémentée dans l'environnement SuperCollider. Elle est composée d'une dizaine de classes qui implémentent ces trois fonctionnalités : un étage de réception des signaux, un étage d'extraction de traits caractéristiques et un interpolateur de préréglages. Elle est sans

2 . Les accéléromètres ne peuvent fournir que deux de ces rotations au même moment. Par exemple, lorsque l'éponge est tenue horizontalement, il est impossible de connaître le lacet. Il faudrait ajouter une boussole à l'éponge pour rendre cette donnée disponible.

Figure 4. Une capture d'écran du plugiciel *Doppler* de la collection *GRM Tools*. Le long potentiomètre horizontal situé au bas de la fenêtre permet d'interpoler entre les préréglages numérotés de 1 à 8.

cesse en évolution. La version actuelle peut être téléchargée sur *github* en suivant ce lien :

http ://github.com/marierm/mmExtensions.

3.3.1. Réception des signaux

La réception des signaux est implémentée dans la classe `Sponge` et permet à SuperCollider d'ouvrir le port sériel approprié et de rendre les données brutes des capteurs disponibles aux autres étages de *mapping*.

3.3.2. Extraction de traits caractéristiques

L'expression *extraction de traits caractéristiques* est une traduction de l'anglais *feature extraction*. Un trait caractéristique est un signal qui peut être déduit ou calculé à partir des signaux bruts des capteurs. Par exemple, il est possible d'extraire de l'éponge le trait caractéristique $pitch_1$ (l'inclinaison de l'accéléromètre numéro un) qui est calculé à partir des axes x et z de l'accéléromètre 1 : $pitch_1 = \arctan\left(\frac{acc_1 x}{acc_1 z}\right)$.

Il est aussi possible d'extraire d'autres traits caractéristiques à partir de traits caractéristiques existants et ainsi de suite. Par exemple, les traits caractéristiques $roll_1$ et $roll_2$ sont obtenus respectivement avec les formules $roll_1 = \arctan\left(\frac{acc_1 y}{acc_1 z}\right)$ et $roll_2 = \arctan\left(\frac{acc_2 y}{acc_2 z}\right)$. Par la suite, on peut obtenir le trait caractéristique $twist$ en soustrayant $roll_2$ de $roll_1$. Ces interdépendances sont gérées automatiquement par la classe `Feature` : si un utilisateur active le trait caractéristique $twist$, les traits caractéristiques $roll_1$ et $roll_2$ seront automatiquement activés.

L'extraction de traits caractéristiques permet de séparer le *mapping* en plusieurs étages, ce qui facilite la conception de *mappings* complexes [16]. De plus, cette stratégie rend disponibles des signaux qui sont plus représentatifs du geste que les signaux bruts des capteurs, ce qui aide à obtenir un contrôle musical.

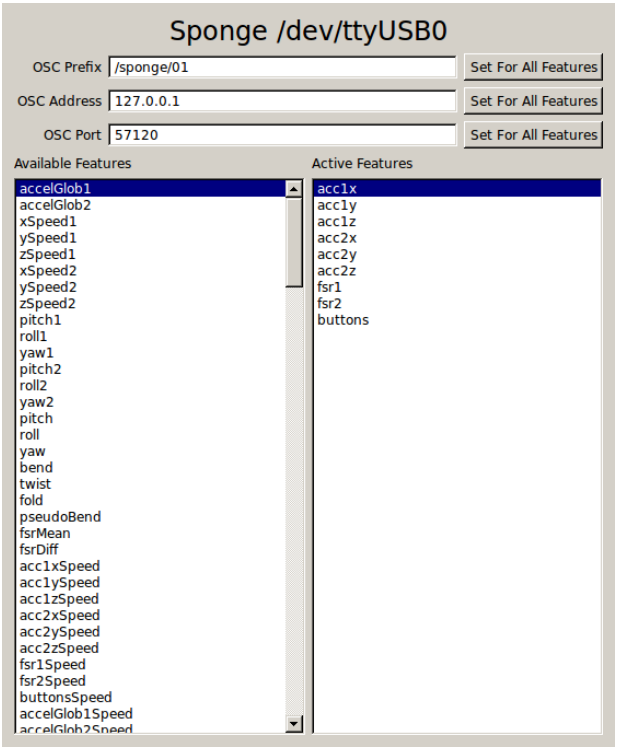


Figure 5. L'interface graphique qui permet d'activer et de désactiver les nombreux traits caractéristiques préprogrammés.

Il existe une multitudes de traits caractéristiques (environ 200) préprogrammés pour l'éponge qui peuvent être activés ou désactivés très rapidement selon les besoins de l'utilisateur. La figure 5 est une capture d'écran de l'interface graphique qui permet de gérer l'activation des différents traits caractéristiques. Les signaux correspondants aux traits caractéristiques deviennent alors disponibles à l'utilisateur. Il est aussi possible de les visionner (figure 7) et de les acheminer vers d'autres applications ou d'autres appareils sous forme de messages OSC (figure 6 et 5).

3.3.3. Interpolation de préréglages

Un système d'interpolation de préréglages permet à l'utilisateur de faire varier une multitude de paramètres en n'agissant que sur un nombre limité de contrôles [3, 1, 14, 4, 10]. De par sa nature, un tel système facilite grandement la conception de *mappings* complexes de type *many-to-many*, ce qui permet aussi de se rapprocher d'un contrôle musical.

Le système d'interpolation de préréglages que j'ai développé prend la forme d'un espace virtuel (à une ou plusieurs dimensions) dans lequel il est possible de disposer une multitude de points représentant chacun un préréglage. Par la suite, l'utilisateur peut déplacer un curseur dans cet espace. La valeur des multiples paramètres de chacun des préréglages varie avec la position du curseur.

Le tableau 1 illustre le fonctionnement d'un interpolateur de préréglages à une dimension qui comporte simplement deux points. Dans cet exemple, deux paramètres

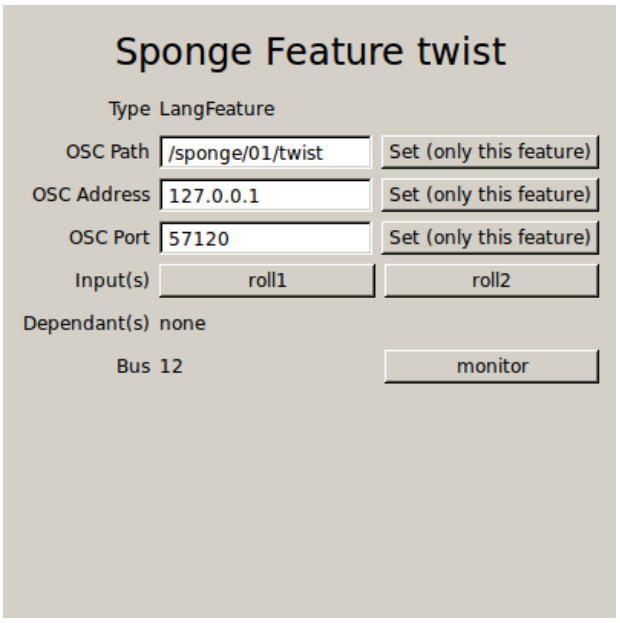


Figure 6. L'interface graphique du trait caractéristique *twist*. L'utilisateur peut choisir le format du message OSC qui sera envoyé.

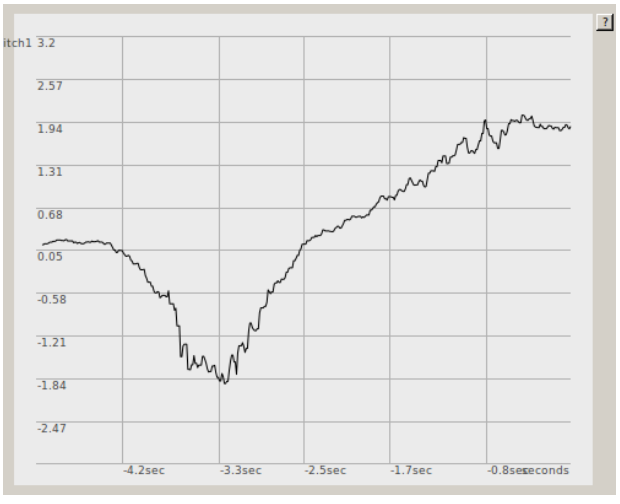


Figure 7. Il est possible de visualiser les signaux des capteurs et des traits caractéristiques.

d'un filtre en cloche varient en fonction de la position du contrôle d'interpolation.

Position du contrôle d'interpolation	Poids des préréglages	Gain	Fréquence centrale
A	A	+12 dB	400 Hz
$\frac{1}{4}$ de course	$\frac{3}{4}A + \frac{1}{4}B$	+10.5 dB	800 Hz
$\frac{1}{2}$ de course	$\frac{1}{2}A + \frac{1}{2}B$	+9 dB	1200 Hz
$\frac{3}{4}$ de course	$\frac{1}{4}A + \frac{3}{4}B$	+7.5 dB	1600 Hz
B	B	+6 dB	2000 Hz

Table 1. Les valeurs que prennent les paramètres d'un filtre cloche en fonction de la position du contrôle d'interpolation.

Les classes `Interpolator` et `PresetInterpolator` implémentent un système d'interpolation de préréglage beaucoup plus complexe et beaucoup plus versatile. Il est possible d'y créer des espaces à un nombre arbitraire de dimensions et d'y placer un nombre infini de préréglages (l'utilisateur n'est limité que par les capacités de son ordinateur). Le nombre de dimensions de l'espace correspond au nombre de signaux de contrôles utilisés (le nombre de capteurs). Par exemple, on peut choisir de connecter un accéléromètre 3D à un interpolateur de préréglages à trois dimensions (une dimension pour chacun des axes de l'accéléromètre). Autrement dit, on navigue dans l'espace 3D à l'aide de l'accéléromètre. Les espaces à plus de trois dimensions sont plus difficile à imaginer, mais il s'agit d'avoir autant de dimensions que de contrôleurs continus.

Pour utiliser l'interpolateur de préréglage avec l'éponge, on peut se créer un espace d'interpolation à huit dimensions auxquelles on connecte les huits signaux continus de l'éponge. Le nombre de paramètres que l'on peut contrôler est complètement indépendant du nombre de dimension.

4. ENJEUX ET DÉFIS

4.1. L'éponge : interface ou instrument ?

Les acteurs importants du domaine des nouvelles interfaces (ou nouveaux instruments) ne s'entendent toujours pas sur la définition exacte de ces deux termes. Pour les besoins de la discussion, j'adopterai la définition suggérée par [9] et illustrée par la figure 9. Un instrument complet est constitué de quatre étages successif : l'interface, le *mapping*, la génération du son et la diffusion. Comme on peut le voir, l'interface n'est en fait que le premier étage d'un instrument, et c'est exactement ce que l'éponge est.

Fait important à noter, sur un instrument de musique numérique, les différents étages sont discrets, alors que sur la majorité des instruments acoustiques, les éléments sont indissociables les uns des autres.

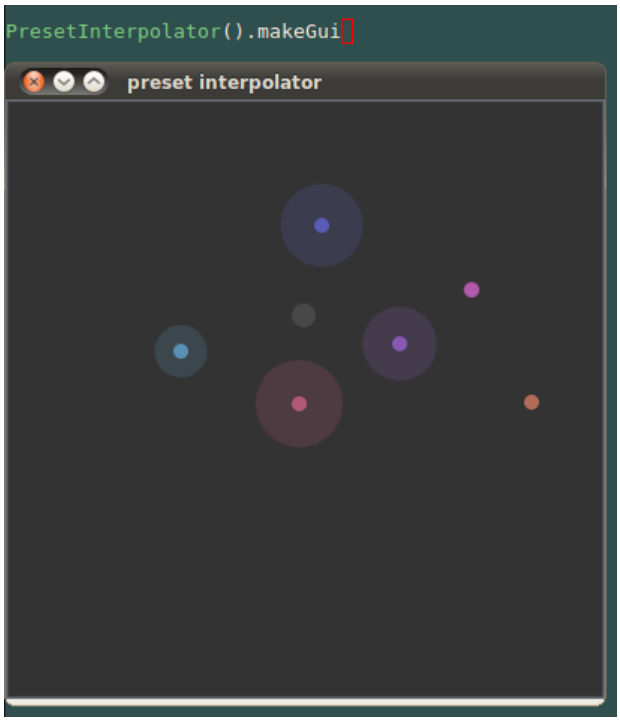


Figure 8. La fenêtre *preset interpolator* permet à l'utilisateur de visualiser et de modifier l'emplacement des préréglages dans l'espaces. Si l'espace comporte plus de deux dimensions, il est évidemment impossible de toutes les visualiser. Cependant, il est possible d'ouvrir plusieurs fenêtre *preset interpolator* et de choisir quelles dimensions seront visualisées dans chacune d'elles. Chacun des points colorés correspond à un préréglage. Le point gris légèrement plus gros que les autres est le curseur. Le poids des points est représenté par des cercles transparents : plus le cercle autour d'un point est grand, plus le point a du poids.

4.2. Établir un lien d'énergie

Plusieurs chercheurs sur les interfaces musicales [12, 7, 11] ont déjà tiré la conclusion que, si l'on désire un rapport geste-son clair, il doit y avoir un transfert de l'énergie du geste vers l'énergie du son. Pour établir ce transfert d'énergie, il est suggéré de faire correspondre la vitesse du geste à l'amplitude du signal [6]. Avec l'éponge et son système d'extraction de traits caractéristiques, il est possible d'avoir rapidement accès aux différentes *vitesses* qui sont en jeu :

- la vitesse de pression (dérivée du signal des capteurs de pressions) ;
- la vitesse de rotation globale de l'éponge sur chacun des axes ;
- la vitesse de pliage et de torsion.

De plus, comme l'accélération est elle-même directement proportionnelle à l'énergie, il est tout à fait cohérent d'affecter l'amplitude d'un son avec les signaux qui proviennent directement des accéléromètres. D'ailleurs, dans la pièce *Clarinette* (cette pièce est discutée à la section 5), les signaux des accéléromètres (filtrés par un passe haut) sont utilisés comme excitateurs pour un synthétiseur

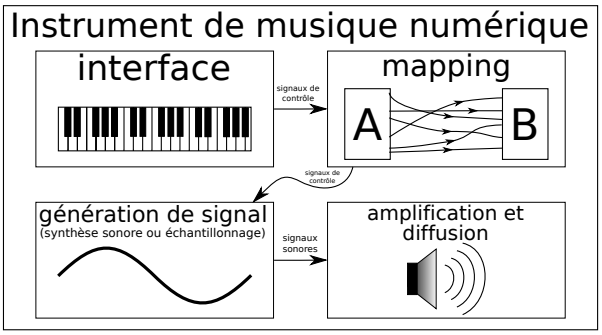


Figure 9. L'instrument de musique est un système complet alors que l'interface ne représente que son premier étage. Cette figure est une adaptation de celle qu'on retrouve dans le livre de [9, p. 3]

à guide d'ondes. Ce *mapping* a beaucoup de succès auprès du public : les gens comprennent intuitivement le lien qui existe entre le geste et le son.

4.3. Gestion de la polyphonie

L'éponge a été conçue comme une interface monophonique mais, rapidement, le besoin de faire de la polyphonie s'est fait sentir. Mis à part la solution évidente qui consiste à utiliser plusieurs éponges, il existe quelques façon d'arriver à jouer plusieurs voix.

La première consiste à utiliser les boutons de l'éponge pour démarrer et arrêter différentes voix. La stratégie utilisée jusqu'à maintenant consiste à attribuer trois états différents à chaque bouton. Voici à quels comportements sont associés chacun des états.

1. Le processus est déclenché : une voix joue et on est en contrôle de ses différents paramètres avec l'éponge.
2. Le processus continue, mais l'éponge ne contrôle plus les paramètres de la voix. La voix est gelée.
3. La voix s'éteint doucement (en fondu).

De cette façon, il est possible d'avoir une voix de polyphonie par bouton, c'est-à-dire sept. Cela fonctionne bien, mais il est très difficile pour l'interprète de se souvenir de l'état de chacune des voix. En réalité, il est impossible de s'en sortir lorsque plus de trois voix dont le timbre est semblable sont utilisées en même temps.

La deuxième façon de faire est très semblable ; il n'y a qu'une exception : au lieu d'utiliser le deuxième état pour geler la voix, on l'utilise pour boucler la voix. Aussitôt que le boucleur de geste décrit à la section 6.1 sera fonctionnel, cette stratégie remplacera la première.

4.4. Composer avec la multidisciplinarité

Être à la fois compositeur, interprète et concepteur d'instrument pose certains problèmes qui peuvent être illustrés par l'exemple suivant.

Imaginons un compositeur-interprète-concepteur qui se retrouve face à un problème musical typique : il n'est pas satisfait du résultat sonore qu'il entend et veut le changer. Trois possibilités s'offrent alors à lui.

D'abord, il peut opter pour l'approche de l'interprète. Dans ce cas, il pratiquera jusqu'à ce qu'il ait développé les habiletés qui lui permettront de jouer la phrase musicale problématique de façon satisfaisante. Il pourrait aussi opter pour l'approche du compositeur, auquel cas il choisira de remettre en question la musique elle-même et de recomposer des phrasés mieux adaptés à l'instrument. Sa dernière option serait l'approche du concepteur. Dans ce dernier cas, il choisira de modifier l'instrument de façon à ce qu'il devienne possible de produire un résultat plus satisfaisant. Cette solution est la plus laborieuse puisqu'elle impose à l'interprète de réapprendre à jouer de l'instrument modifié.

Un instrument performant et expressif doit se plier aux besoins de l'interprète et du compositeur, mais le développement continuel de l'instrument empêche de fixer les oeuvres et, pire encore, de rejouer des oeuvres composées pour une version antérieure de l'instrument.

Découvrir et développer des stratégies qui permettront de conjuguer développement continuel et pérennité des oeuvres est une partie importante de cette recherche. Pour l'instant, des systèmes des gestions de versions comme Subversion ou Git (très utilisés par les programmeurs) sont utilisés pour pouvoir facilement jongler avec les multiples versions d'une pièce. Cette solution est très pratique pour le gestion de la partie logicielle, mais n'est d'aucune utilité pour gérer différentes versions d'une interface.

4.5. La question de la pérennité

La plupart du temps, en musique électroacoustique, la partition ne sert que de support pour l'analyse ou d'aide à la diffusion. Comme il s'agit traditionnellement d'un art des sons fixés, cela ne pose pas de problème : la pérennité de l'oeuvre est assurée par son support. Mais lorsqu'on tombe dans le domaine des arts d'interprétation, une oeuvre n'est considérée pérenne qui si elle peut être réinterprétée. Donc, pour assurer une longue vie à une oeuvre pour éponge, on doit d'abord s'assurer que les appareils et logiciels utilisés pourront être remplacés au fur et à mesure que la technologie évolue. On doit aussi veiller à ce qu'un interprète puisse apprendre et rejouer l'oeuvre.

Il est possible de remplir la première condition en documentant clairement les processus utilisés de façon à ce qu'il soit possible de réimplémenter les mêmes algorithmes en utilisant la technologie du futur. Il est aussi possible d'utiliser des technologies que l'on croit plus durables que d'autres comme par exemples des logiciels dont le code source est ouvert.

Remplir la deuxième condition est délicat. On pourrait croire que la partition est la solution, mais ce n'est pas si simple. Un des objectifs de cette recherche est de permettre l'interprétation en exploitant le médium électroacoustique. Hors, il n'existe pas de système de notation standard pour ce genre de musique. Il est certes possible d'en concevoir un, mais basé sur quoi ? Sur la notation traditionnelle ? Sur une notation du geste comme celle de Laban pour la danse ?

Thierry De Mey, pour sa pièce *Light Music* [3] , a opté pour une partition hybride. Certaines sections utilisent la notation musicale traditionnelle, alors que d'autres emploient des dessins qui évoquent les gestes que l'interprète doit faire. Cette seule partition ne suffirait pas à assurer la pérennité de cette oeuvre. D'ailleurs, le compositeur et son équipe ont dernièrement organisé un stage de formation dont le but était de former des interprètes pour cette oeuvre.

Pour l'instant, développer un système de notation rigoureux et systématique pour les oeuvres pour éponge n'est pas une priorité. Pour assister le travail sur la structure des pièces, un simple séquenceur est utilisé. Après avoir été enregistrées, des performances sont retravaillées dans le logiciel. Par la suite, j'essaie de rejouer à l'éponge la version retravaillée. Si la version est injouable (ce qui est habituellement le cas) je recommence.

Dans ce procédé, le logiciel constitue un substitut à la partition, mais uniquement pour le travail de composition ; il ne permet aucunement la transmission des oeuvres. Pour permettre une éventuelle transmission d'une oeuvre, une stratégie impliquant des captations vidéo de bonne qualité et de différents points de vue serait choisie plutôt qu'un système de notation.

5. LA COMPOSITION

5.1. *Cymbale*, la pièce morte

Cymbale avait été composée pour la toute première version de l'éponge, celle qui était construite à partir d'une interface Infusion Systems BlueTooth [8]. Cette technologie ne s'est pas avérée adéquate pour ce type d'interface et, peu de temps après la création de la pièce, l'éponge 2.0 a vu le jour. Adapter *Cymbale* à la nouvelle version de l'éponge aurait représenté un travail considérable et, comme elle n'était pas très satisfaisante esthétiquement, la pièce fut abandonnée.

La décision a été consciente, mais la mort de *Cymbale* n'était pas voulue. Cet événement m'a appris à quel point les pièces écrites pour nouvelles interfaces sont fragiles.

5.2. *Clarinette* et ses descendants

Clarinette a un destin différent : elle est toujours jouée et elle évolue sans cesse. En fait, elle évolue à un point tel qu'il est difficile de considérer qu'il s'agit encore de la même pièce !

Clarinette est donc une pièce laboratoire. À chaque nouvelle mouture, au fur et à mesure que l'éponge offre de nouvelles possibilités, de nouveaux éléments se greffent. Au départ, il n'y avait que des enregistrements de clarinette traités par granulation. Ensuite, des sons de synthèse

3 . *Light Music* est une pièce pour « chef d'orchestre » et dispositif électronique. Les gestes de l'interprète sont suivis par des caméras vidéos et affectent le son

par guide d'onde se sont ajoutés. Parallèlement à cela, les stratégies de *mapping* se sont raffinées. L'interpolateur de préréglage s'est perfectionné et est devenu multidimensionnel, ce qui permet d'exploiter une plus grande variété de gestes et d'avoir plus de précision. Des boutons ont été ajoutés, ce qui permet d'avoir un contrôle sur la polyphonie, de déclencher des événement et de jouer avec des hauteurs discrètes. Enfin, des filtres formantiques et de la distorsion ont été intégrés.

Si on ajoute à cela le fait que la structure n'est toujours pas fixée, il n'est pas très étonnant que les différentes versions de *Clarinette* ne se ressemblent pas beaucoup. Pour cette raison, elle porte toujours un sous-titre. Il existe trois versions importantes de la pièce :

Clarinette (2009) La première version jouée sur l'éponge originale [8]. Les matériaux sonores étaient constitués uniquement de sons de clarinette granulés.

Struggling (2010) Cette version était jouée sur l'éponge 2.0 (identique à la version actuelle, mais sans boutons) et incorporait de la synthèse par guide d'onde. Pour la première fois, il était possible de jouer des percussions sur l'éponge sans problèmes de latence.

Albino Butterfly (2011) C'est la première version à exploiter les boutons. Ils permettent de gérer jusqu'à quatre voix de polyphonie. La distorsion et les filtres formantiques ont été intégrés.

Clarinette devrait se fixer avant la fin de 2012, mais pour l'instant, elle reste une pièce laboratoire grâce à laquelle il est possible d'expérimenter avec l'éponge devant public.

6. DÉVELOPPEMENTS FUTURS

6.1. Boucleur de geste

Depuis les expérimentations de [13] sur le sillon fermé, le concept de la boucle audio est bien connu : l'idée est simplement de répéter cycliquement un signal audio.

Le boucleur de geste est un système pratiquement identique sauf qu'il est conçu pour boucler des signaux de contrôle plutôt que des signaux audio.

Cette nuance peut sembler anodine, mais si on y regarde de plus près, on constatera que choisir de boucler le geste plutôt que le son a un impact important.

6.1.1. Concept

Pour boucler un geste, il faut boucler un ou plusieurs signaux de contrôle. Il peut s'agir de signaux obtenus directement des capteurs ou encore de traits caractéristiques qui ont été préalablement extraits. Les signaux ainsi bouclés peuvent alors affecter les paramètres qui étaient contrôlés avec l'éponge. L'interprète, qui n'a plus à contrôler ces paramètres, peut alors décider de créer une nouvelle voix de polyphonie en se faisant accompagner par la boucle.

Un tel système n'offre pas d'avantages majeurs par rapport au boucleur de son traditionnel. Ce qui est différent, c'est qu'il devient aussi possible de boucler un nombre limité de paramètres et de garder le contrôle manuel d'autres paramètres. Par exemple, si on enregistre une boucle dans laquelle les paramètres *fréquence*, *amplitude* et *amplitude du vibrato* varient, on peut boucler les deux premiers paramètres et garder le contrôle manuel l'*amplitude du vibrato*.

Il devient aussi possible d'appliquer le concept du mode *trim* [4] qu'on retrouve dans les séquenceurs audionumériques comme ProTools ou Digital Performer. En effet, plutôt que de prendre le contrôle d'un paramètre, on peut le boucler et venir l'altérer en temps réel. De cette façon, on répète le geste musical enregistré, mais on fait finement varier la boucle.

6.1.2. Intégration avec l'interpolateur de préréglages

Le boucleur de geste en lui-même n'aide aucunement à établir un contrôle musical. Son utilité est à un autre niveau : il permet à un interprète de gérer plusieurs voix de polyphonie avec une interface qui, à prime abord, semble suggérer la monophonie.

Mais si on veut se rapprocher du contrôle musical, on peut l'utiliser de pair avec l'interpolateur de préréglage : plutôt que de boucler directement les paramètres, on peut boucler des trajectoires dans l'espace multidimensionnel. De cette façon, une seule boucle affecte une multitude de paramètres. On peut alors utiliser le mode *trim* pour venir modifier en temps réel la trajectoire bouclée.

6.1.3. Implémentation

Au moment de l'écriture de ce document, le boucleur de geste n'est pas encore utilisable. Les fonctionnalités de bases sont implémentées dans les classes `Looper` et `FeatureLooper`, mais l'intégration avec l'interpolateur de préréglages et l'éponge elle-même n'est pas encore codée.

6.2. Projet de pièces finales

L'objectif est d'avoir quatre pièces pour éponge. Le but est d'écrire des oeuvres dont la structure est claire et perceptible dès une première écoute. Le principe de tension-détente et la recherche d'un équilibre entre redondance et originalité guideront l'écriture.

Les premières pièces pour éponge (*Cymbale* et *Clarinette*) étaient plutôt minimalistes et *ambient*. Cela vient bien entendu d'un intérêt pour ce genre, mais les contraintes techniques liées à l'utilisation d'une nouvelle interface y étaient aussi pour beaucoup. Maintenant que l'éponge a atteint une certaine maturité et que les temps de latence ont été grandement réduits, il est désormais plus facile de jouer avec des matériaux plus articulés et plus dynamiques. Les nouvelles pièces comporteront des sections rythmiques dans lesquelles la pulsation sera l'ancrage du discours musical.

4 . Le mode *trim* est un mode d'enregistrement d'automatisations qui additionne le geste enregistré à la courbe qui est déjà là.

6.3. Interface, *mappings* et pièces : des modules assemblables

À écrire de la musique pour une nouvelle interface, on est forcé de se demander si le *mapping* utilisé est associé à l'interface ou à la pièce. Lorsqu'un *mapping* est associé à l'interface, on se retrouve avec un instrument de musique numérique pour lequel on peut écrire plusieurs pièces. À l'inverse, si le *mapping* est associé à une pièce, on obtient une entité *pièce-mapping-interface* monolithique.

Dans le travail à venir, ces deux avenues seront explorées : il y aura deux pièces pour un même instrument et une autre pièce composée pour un *mapping* complètement différent.

De plus, une pièce pour deux éponges sera écrite. En plus de permettre une polyphonie plus évoluée, cette avenue permettra d'explorer les possibilités d'interaction avec d'autres musiciens.

7. CONCLUSION

Le travail fait au cours des dernières années a permis d'amener l'éponge à un certain niveau de maturité. L'interface est devenue à la fois plus robuste et plus efficace, et la couche logicielle rend maintenant la conception de *mappings* complexes beaucoup facile et rapide. Parallèlement à l'évolution de l'éponge, il y a eu mon évolution personnelle : j'ai énormément progressé comme interprète… comme épongiste.

Il y aura toujours quelque chose à améliorer sur l'éponge, mais il semble que, pour la première fois depuis quatre ans, mon outil est enfin prêt et que je peux maintenant composer et jouer de la musique.

8. REMERCIEMENTS

Merci à Prof. Jean Piché, Annie Lalancette, Dr. Garth Paine et à Georges Forget. Nos échanges me nourrissent énormément. Sans votre support, l'éponge laverait encore de la vaisselle.

9. REFERENCES

[1] Ross Bencina. The metasurface : applying natural neighbour interpolation to two-to-many mapping. In *Proceedings of the 2005 conference on New interfaces for musical expression*, pages 101–104. National University of Singapore, 2005.

[2] Christopher Dobrian and Daniel Koppelman. The'E'in NIME : musical expression with new computer interfaces. In *Proceedings of the 2006 conference on New interfaces for musical expression*, pages 277–282. IRCAM—Centre Pompidou, 2006.

[3] Adrian Freed, John MacCallum, Andrew Schmeder, and David Wessel. Visualizations and Interaction Strategies for Hybridization Interfaces. In *NIME '10 : Proceedings of the 2010 conference on New interfaces for musical expression*, number Nime, pages 343–347, Sydney, 2010.

[4] Camille Goudeseune. Interpolated mappings for musical instruments. *Organised Sound*, 7(02) :85–96, 2003.

[5] Andy Hunt and Marcelo M. Wanderley. Mapping performer parameters to synthesis engines. *Organised Sound*, 7(02) :97–108, 2002.

[6] Andy Hunt, Marcelo M. Wanderley, and Ross Kirk. Towards a Model for Instrumental Mapping in Expert Musical Interaction. In *Proceedings of the 2000 International Computer Music Conference*, pages 209–212, 2000.

[7] Andy Hunt, Marcelo M Wanderley, and Matthew Paradis. The importance of parameter mapping in electronic instrument design. In *NIME '02 : Proceedings of the 2002 conference on New interfaces for musical expression*, pages 1–6, Singapore, Singapore, 2002. National University of Singapore.

[8] Martin Marier. The Sponge : A Flexible Interface. In *NIME '10 : Proceedings of the 2010 conference on New interfaces for musical expression*, pages 356–359, Sydney, 2010.

[9] Eduardo R. Miranda, Marcelo M. Wanderley, and Ross Kirk. *New digital musical instruments : control and interaction beyond the keyboard*. AR Editions, Inc., Middleton, 2006.

[10] Ali Momeni and David Wessel. Characterizing and controlling musical material intuitively with geometric models. In *Proceedings of the 2003 conference on New interfaces for musical expression*, pages 54–62. National University of Singapore, 2003.

[11] Garth Paine. Towards Unified Design Guidelines for New Interfaces for Musical Expression. *Organised Sound*, 14(02) :142–155, 2009.

[12] Joel Ryan. Some remarks on musical instrument design at STEIM. *Contemporary Music Review*, 6(1) :3–17, 1991.

[13] Pierre Schaeffer. *Traité des objets musicaux*. Éditions du Seuil, Paris, France, 1ère edition, 1966.

[14] Martin Spain and Richard Polfreman. Interpolator : a two-dimensional graphical interpolation system for the simultaneous control of digital signal processing parameters. *Organised Sound*, 6(02) :147–151, February 2002.

[15] M M Wanderley, B W Vines, N Middleton, C McKay, and W Hatch. The musical significance of clarinetists' ancillary gestures : An exploration of the field. *Journal of New Music Research*, 34(1) :97–113, 2005.

[16] M.M. Wanderley, Norbert Schnell, and Joseph Rovan. Escher-modeling and performing composed instruments in real-time, 1998.

ATELIER : INTRODUCTION AU LANGAGE DE PROGRAMMATION FAUST

Yann ORLAREY
GRAME, Centre national de création musicale
9 rue du Garet
69202 Lyon,
France,
orlarey@grame.fr

RÉSUMÉ

FAUST est un langage de programmation fonctionnel synchrone spécialement conçu pour le traitement du signal et la synthèse de sons en temps réel. L'objectif de l'atelier est de proposer une introduction simple au langage et à ses possibilités, notamment en relation avec les autres environnements musicaux. Les participants qui le souhaitent peuvent venir avec leur machine.

1. PRÉSENTATION

FAUST [1] est un langage de programmation spécialisé, conçu pour décrire de manière concise des algorithmes de synthèse et de traitement du son. Un programme FAUST décrit un processeur de signaux, c'est à dire une fonction, au sens mathématique du terme, qui prend des signaux en entrée et produit des signaux en sortie.

L'une des caractéristiques de FAUST, contrairement aux autres langages musicaux, est d'être entièrement compilé. On peut donc tout à fait utiliser FAUST à la place de C pour écrire par exemple des plugins audio. Les techniques de compilation mise en oeuvre sont très optimisées et permettent de générer du code de qualité, dont l'efficacité est généralement comparable à du code C écrit à la main.

Le système d'architecture de FAUST [2] facilite le déploiement des programmes et permet de générer, à partir d'un même source, du code pour les principales plateformes audio : MaxMSP, VST, Puradata, Csound, Supercollider, etc.

FAUST est un langage textuel. Sa syntaxe est basée sur l'idée de *composition* de processeurs de signaux. Ainsi par exemple si `A` et `B` sont deux processeurs de signaux, `(A:B)` représente le processeur de signaux obtenu en branchant les sorties de `A` sur les entrées correspondantes de `B`. Tandis que `(A,B)` représente la mise en parallèle de `A` et `B`.

Un programme FAUST est constitué d'un ensemble de définitions dont celle du mot clef `process`, l'équivalent de `main()` en C.

Par exemple le programme suivant comporte 3 définitions et décrit un générateur de bruit blanc dont le niveau est contrôlé par un réglage de volume dont la valeur par défaut est 0, qui prend des valeurs entre 0 et 1 avec un pas de variation de 0.01.

```
noise = random / 2147483647.0;
random  = +(12345) ~ *(1103515245);
process = noise * hslider("volume", 0, 0,
    1, 0.01);
```

Il est possible d'associer, en parallèle de l'interface graphique, d'autres interfaces de contrôles comme OSC ou HTTP. L'exemple qui suit est une variante du précédent où l'on indique que le slider de volume est contrôlé également par la réception du message OSC `/accxyz` ce qui permet de piloter ce slider à distance via un smartphone et l'application TouchOSC.

```
noise = random / 2147483647.0;
random  = +(12345) ~ *(1103515245);
process = noise * hslider("volume[osc:/
    accxyz/0␣-10␣10]", 0, 0, 1, 0.01);
```

La façon la plus simple de tester FAUST est d'utiliser le compilateur en ligne `http://Faust.grame.fr`. Depuis la page d'accueil cliquez sur *Online Examples*. Après avoir fermé la fenêtre de bienvenue choisissez dans la rubrique *Effects* le programme *freeverb*. Cliquez ensuite sur l'onglet *C++ code* pour choisir l'architecture souhaitée, par exemple *VST*. Enfin allez sur l'onglet *Exec File* pour déclencher la compilation et récupérer le plugin VST prêt à l'emploi.

Le site contient également toutes les informations necessaires pour téléchargez FAUST et l'installer sur une machine Linux ou MacOSX.

2. REFERENCES

[1] Orlarey, Y., Fober, D. et Letz, S. "An algebra for block diagram languages", *Proceedings of the International Computer Music Conference (ICMA)*, Gothenburg, Suède, 2002.

[2] Fober, D., Orlarey, Y. et Letz, S. "FAUST architecture design and OSC support", *Proceedings of the Conference on Digital Audio Effects (DAFx-11)*, IRCAM, Paris, France, 2011.

ATELIER IANNIX

Guillaume Jacquemin
Association IanniX
guillaume@iannix.org

Thierry Coduys
Association IanniX
thierry@iannix.org

IANNIX

IanniX est un séquenceur graphique open source, inspiré des travaux de Iannis Xenakis et destiné à la création numérique. Le logiciel propose une écriture polytemporelle du temps d'événements statiques et dynamiques vers un environnement dédié (PureData, SuperCollider, CSound…).

À l'aide d'une palette d'objets fondamentaux que sont les *triggers (événements)*, les courbes *(trajectoires dans l'espace)* et les curseurs *(progression dans le temps)*, IanniX permet une représentation graphique et interactive du temps dans l'espace 3D et assure un échange bidirectionnel via plusieurs protocoles de communication, dont l'Open Sound Control.

Les partitions IanniX s'écrivent grâce à une interface graphique ou à l'aide de code JavaScript.

METHODOLOGIE

L'atelier se déroule en deux temps :

- découverte pas-à-pas (45mn) : l'équipe présente IanniX à l'aide d'une série de manipulations fondamentales et explique aux participants les détails et fonctionnements d'une performance ;
- mise en œuvre individuelle (45mn) : chaque participant est invité à exploiter IanniX avec un outil audionumérique de son choix.

L'atelier s'adresse à un public connaissant les bases de l'Open Sound Control.

CONTENU DETAILLE

- Installation et prise en main
  - Installation de IanniX sur les ordinateurs des participants
  - Création d'un séquenceur « traditionnel » (courbe + curseur + trigger)
- Principe de polytemporalité
  - Les différentes échelles de temps
  - Multi-curseurs
  - Motifs temporels
- Communication avec un logiciel tiers
  - Fonctionnement des messages
  - Interfaces physiques
  - Personnalisation d'un message
  - Réception de messages
- Scripts génératifs
  - Template d'un script
  - Syntaxe JavaScript et API IanniX
  - Génération d'une partition
  - Exemples mathématiques divers
- Ouvertures
  - Écriture récursive via l'interface virtuelle
  - Matriçage des messages IanniX
- Mise en œuvre personnalisée
  - Vérification des protocoles
  - Découverte des exemple de synthèse sonore ou/et exploitation personnelle de IanniX

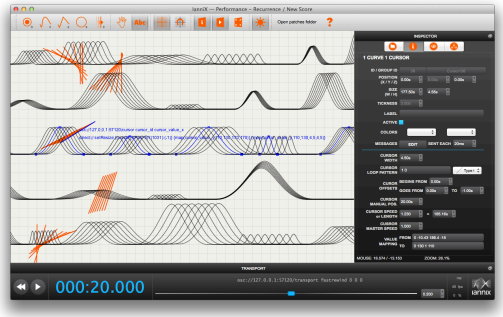


Figure 1 : Exemple de manipulation avec IanniX

RECOMMANDATIONS TECHNIQUES

Le matériel suivant constitue le minimum requis pour réaliser l'intégralité de l'atelier :

- Un ordinateur portable :
  - Mac OS X, 10.5 minimum, processeur Intel
  - Linux *(installation au préalable par le participant)*
  - Windows XP ou supérieur
- Un logiciel client :
  - compatible Open Sound Control
  - proposition de logiciels (non exhaustif) : http://iannix.org/fr/links.php

ENCADRANTS

Thierry Coduys est le chef de projet IanniX. Artiste polyvalent, musicien, spécialiste des nouvelles technologies, il s'intéresse en particulier aux projets liant interactivité et art contemporain.

Guillaume Jacquemin est le développeur de IanniX depuis mai 2011. D'une formation initiale d'ingénieur en systèmes embarqués et titulaire d'un master en optimisation de systèmes, il fonde en 2009 la société buzzing light opérant dans le design d'interactions.

EANALYSIS : AIDE A L'ANALYSE DE LA MUSIQUE ELECTROACOUSTIQUE

Pierre Couprie
MTI, Université De Montfort (Leicester) / OMF-MINT, Université Paris-Sorbonne (Paris)
pierre.couprie@paris-sorbonne.fr

RÉSUMÉ

Cet article présente le logiciel EAnalysis, logiciel d'aide à l'analyse musicale dans le domaine de la musique électroacoustique. Le développement, qui est en cours de réalisation et ce poursuivra jusqu'en octobre 2013, s'inscrit dans le projet de création d'une boîte à outils pour l'analyse de la musique électroacoustique à l'Université De Montfort de Leicester (Grande-Bretagne). EAnalysis a pour objectif d'expérimenter de nouvelles formes de représentations graphiques et de nouvelles méthodes d'analyse musicale pour cet art encore très jeune dont les pratiques ne cessent de s'enrichir de jours en jours. A terme, il permettra de manipuler des sources variées (audio, vidéo, image, données extraites d'autres logiciels, etc.) à travers une interface simple, modulaire et intuitive. Il offrira aussi la possibilité de partager facilement ses analyses et de développer ses propres outils analytiques.

1. INTRODUCTION

Le logiciel EAnalysis[1] s'inscrit dans le projet de recherche *New Multimedia Tools for Electroacoustic Music Analysis* mené par le centre de recherche *Music, Technology and Innovation*[2] de l'université De Montfort de Leicester (Grande Bretagne). Le projet est placé sous la direction des professeurs Simon Emmerson et Leigh Landy et est supporté par l'AHRC[3]. Actuellement, deux des premières étapes sont disponibles : le site web communautaire OREMA[4] conçu par Michael Gatt et le logiciel EAnalysis[5]. OREMA comme EAnalysis sont en cours de réalisation et aboutiront à une version finale fin 2013. Un ensemble de publications et un DVD compléteront le projet. D'ors et déjà, les enregistrements vidéo de deux premières journées de recherche sont disponibles sur le site OREMA. Ces journées ont permis de préciser le champ d'étude du projet et d'explorer les pratiques d'analyse et de réception liées au genre électroacoustique.

EAnalysis est basé sur iAnalyse qui va devenir iAnalyse Studio[6]. J'ai eu l'occasion de présenter iAnalyse lors de JIM précédentes [5]. Comme son grand frère EAnalysis est un logiciel d'aide à l'analyse musicale, mais il est spécialisé dans le domaine de la musique électroacoustique.

L'objectif de ce projet est d'expérimenter de nouvelles méthodes de représentations et d'analyse musicale à travers une interface simple, intuitive et ouverte. Ces nouvelles directions de recherche ont déjà été présentées en 2007 lors d'une conférence EMS [4]. Même s'il partage quelques points communs avec le logiciel Acousmographe [10] ou avec des expérimentations plus récentes comme l'Acousmoscribe [8], EAnalysis s'en différencie sur son architecture, ses outils adaptés au travail du musicologue ou les processus de description mis en jeu.

Je ne reviendrai pas ici sur l'intérêt que je développe depuis plusieurs années sur la représentation graphique dans le cadre de l'analyse de la musique électroacoustique [6].

Dans la première partie de cet article, je présente le contexte du logiciel. Dans une deuxième partie, je détaille l'architecture de EAnalysis ainsi que les différents éléments d'interface. Enfin, la dernière partie me permet d'envisager les développements futurs de ce projet.

2. AUX ORIGINES DE EANALYSIS

2.1. La musique électroacoustique

Les pratiques musicales en électroacoustique se développent très rapidement au point qu'il semble parfois vain de vouloir en définit le champ. Plusieurs définitions existent[7] et certains chercheurs comme Leigh Landy [13] tentent de définir un cadre en cherchant à comprendre les relations qui peuvent exister entre des musiciens venant d'univers musicaux variés et pratiquant une musique souvent classée à la marge. Ce champ musical, finalement très récent et surtout extrêmement mobile, oblige le chercheur à penser

[1] Le logiciel EAnalysis sera disponible en version bêta courant mars-avril 2012 et dans une première version finale en septembre 2012. Il sera compatible avec le système Macintosh OS10.6 ou supérieur.
[2] http://www.mti.dmu.ac.uk.
[3] *Arts & Humanities Research Council* : http://www.ahrc.ac.uk.
[4] *Online Repository for Electroacoustic Music Analysis* : http://www.orema.dmu.ac.uk.
[5] http://eanalysis.pierrecouprie.fr.
[6] La version de iAnalyse Studio (numérotée 4) sera disponible en septembre 2012. Elle contiendra une suite de différents logiciels adaptés à l'analyse de la musique écrite.
[7] Définitions à trouver sur le site EARS (*ElectroAcoustic Resource Site*) : http://www.ears.dmu.ac.uk/spip.php?rubrique125.

l'analyse d'une manière différente de la musique instrumentale. En effet, l'absence de support, la complexité du matériau sonore, l'usage des espaces interne et externe, le lien étroit entre les outils et le résultat musical, l'intégration du lieu dans le processus de création, la frontière désormais inexistante entre le sonore et le musical ou le mélange avec d'autres formes artistiques sont en train de bouleverser la musique et la musicologie.

Ces bouleversements nécessitent d'avoir non seulement de nouveaux outils théoriques, mais aussi de nouveaux outils d'analyse. Toutefois, ces derniers doivent s'ancrer dans une pratique de l'analyse musicale rigoureuse et scientifique.

2.2. L'analyse musicale

L'activité d'analyse musicale est communément décomposée en deux étapes : la description et l'interprétation.

2.2.1. La description du matériau

La première étape consiste à segmenter le matériau à analyser en unités et à collecter les informations sur ces unités. Ces dernières peuvent être situées à un niveau particulier ou sur plusieurs niveaux : des unités les plus fines aux structures les plus grandes. Elles peuvent aussi être adjacentes ou non comme dans le cas de l'analyse de saillances ou de fonctions musicales. Il existe plusieurs outils théoriques permettant d'analyser ces unités depuis la typomorphologie de Pierre Schaeffer [15] jusqu'à la spectromorphologie de Denis Smalley [17] en passant par la grille de langage de Simon Emmerson [9] ou les fonctions de Stéphane Roy [14]. Durant cette première étape, un logiciel tel que l'Acousmographe est très utile, car il permet de repérer ces unités en alliant la visualisation de représentations physiques du son à l'écoute et de les marquer sous la forme de repères ou de symboles graphiques. Malheureusement, cette étape est assez complexe, car elle demande une bonne culture et une certaine expérience de l'analyse musicale. Ce sont probablement les raisons qui font que l'analyse d'œuvres électroacoustique n'est pratiquée que par des compositeurs, à quelques exceptions près. L'Acousmographe ou d'autres logiciels parfois utilisés dans l'analyse de musiques électroacoustiques ne proposent jamais d'aide sur l'analyse musicale elle-même. L'apprenti analyste doit alors explorer les articles existants qui eux-mêmes ne proposent généralement que certains résultats de l'analyse et non la démarche de l'analyste lui-même. Il est donc très difficile pour le non-spécialiste d'aborder cette discipline. Un des objectifs d'EAnalysis et du projet *New Multimedia Tools for Electroacoustic Music Analysis* est de combler en partie ce manque.

2.2.2. L'interprétation des données

La deuxième étape consiste à interpréter les données recueillies lors de la description et proposer un certain nombre de résultats sur la forme, les structures, les éléments liés au compositeur (style, démarche artistique, positionnement par rapport à d'autres œuvres, etc.), la réception de l'œuvre ou encore le rapport à des éléments extramusicaux. Cette courte liste n'est évidemment pas exhaustive. Cette étape se traduit généralement par l'écriture d'un texte accompagné, dans le meilleur des cas, par une représentation graphique temporelle, devenue maintenant un classique. Ce type de représentation graphique est très utile au lecteur, car elle permet de repérer les éléments présentés dans le texte sur le plan temporel, voire de les écouter lors d'une publication multimédia. Toutefois, ce type de représentation influence de plus en plus la manière dont les analyses sont présentées : elles tendent à décrire seconde après seconde ce qu'il se passe dans l'œuvre. Un autre objectif d'EAnalysis est de proposer ou de provoquer d'autres modes de représentations afin d'enrichir les analyses en expérimentant de nouveaux types de recueils de données et d'interprétation dans une démarche pédagogique.

2.2.3. Le partage entre chercheurs

Ces deux étapes analytiques sont complétées par un ensemble de techniques qui permettent au musicologue d'améliorer ou d'enrichir sa pratique de l'analyse musicale. Ici encore, EAnalysis propose de nouvelles techniques au chercheur. Celui-ci pourra élaborer sa propre grille de description du matériau musical accompagnée d'une aide et d'exemples musicaux afin de la partager avec d'autres chercheurs. Cette partie ouverte de l'interface de EAnalysis lui permettra par exemple de mutualiser sa propre interprétation de la typomorphologie de Pierre Schaeffer, une version de la spectromorphologie adaptée à un type particulier de musique électroacoustique ou une nouvelle grille de fonctions musicales alliant symboles graphiques et paramètres musicaux.

2.3. Pour qui ? (1/2)

Enfin, peut-être est-il temps de revenir à l'une des premières questions que nous nous sommes posées lors de la phase préparatoire au projet. A qui s'adresse ce logiciel ? Pour le moment, j'ai parlé de musicologues spécialistes ou débutants. Mais EAnalysis a aussi pour ambition de s'ouvrir à d'autres publics, par exemple les enseignants ou les enfants. Ainsi, dans une dernière phase, il est prévu de développer une interface simplifiée accompagnée d'outils de dessin ludiques pour un jeune public. Ce logiciel pourra alors être utilisé afin d'initier les enfants à l'analyse musicale ou travailler l'écoute.

EAnalysis permettra aussi de partager facilement des projets d'analyse en évitant les problèmes liés de droit

d'auteur. L'utilisateur pourra ainsi exporter son projet sans le média et sans le sonagramme. Le destinataire devra se procurer le média et le logiciel recalculera le sonagramme avec les paramètres choisis par l'auteur de l'analyse.

Ces deux exemples montrent l'ouverture de EAnalysis à de nouveaux publics.

3. EANALYSIS : CONCEPTS ET ARCHITECTURE

3.1. De l'idée au logiciel

La partie précédente m'a permis de situer EAnalysis par rapport aux autres logiciels utilisés dans l'analyse de la musique électroacoustique. A l'origine de ces réflexions se trouve un ensemble d'idées déjà développées dans certaines de mes conférences ou testées à travers des versions expérimentales de iAnalyse.

L'idée essentielle est de déconnecter le rendu graphique de l'analyse qui en est à la source. Cette idée reprend simplement celle des feuilles de style. Une des grandes difficultés lorsque l'on réalise une représentation graphique est de pouvoir expérimenter des directions différentes. Malheureusement aucun logiciel ne permet de modifier rapidement plusieurs paramètres graphiques. En effet, l'analyste réalise sa représentation en dessinant sur une vue et les paramètres graphiques sont fixés une fois pour toutes. Pour modifier la représentation, il doit reprendre les formes graphiques une par une afin de modifier leurs paramètres. EAnalysis contient une couche supplémentaire : chaque paramètre graphique peut être associé à un paramètre analytique explicite (intensité, grain, espace, etc.) ou neutre (un simple mot-clé, un texte, un nombre, etc.). La correspondance entre les paramètres graphiques et ces paramètres analytiques est enregistrée au niveau local sur chaque graphique. Une feuille de style permet de créer de nouvelles règles de liens entre les paramètres analytiques et les paramètres graphiques. Ainsi, il devient très simple de modifier en profondeur une représentation graphique sans toucher aux paramètres de la forme graphique. Ce système permet aussi de générer des types de représentations différentes (animation, courbes graphiques, visualisation hors temps, etc.). De plus, l'utilisateur a aussi la possibilité d'analyser en représentant sans trop se soucier des paramètres analytiques et de les préciser ensuite (ou pas) ou alors de travailler la segmentation de l'œuvre à l'aide des paramètres analytiques et de se soucier ensuite de la représentation qui en résultera.

La deuxième idée est de fournir à l'utilisateur une aide dans son analyse à travers deux systèmes. D'une part, EAnalysis contient une bibliothèque d'*analytic events*[8] reprenant différents éléments issus de travaux de recherche sur la théorie musicale (objet sonore, spectromorphologie, fonctions musicales, unités sémiotiques temporelles, etc.). Cette bibliothèque contient aussi des éléments d'explication, des liens vers certaines pages web et des exemples audio ou vidéo afin d'aider l'utilisateur. De plus, cette bibliothèque est ouverte, elle peut ainsi être modifiée et complétée par l'analyste afin d'être exportée et partagée avec d'autres utilisateurs. Enfin, EAnalysis contiendra à terme un système expert dont le rôle sera d'aider le chercheur débutant à préciser son analyse à travers un ensemble de questions et d'exemples audio.

La troisième idée, déjà entrevue précédemment, consiste à expérimenter de nouvelles formes de représentation en rompant avec la traditionnelle vue temps/fréquence. Une des problématiques de la représentation graphique est la limitation des dimensions et par conséquent des fonctions ou paramètres musicaux représentés. En effet, une représentation graphique typique en deux dimensions ne permet de représenter que trois ou quatre paramètres en simultanés[9]. En ajouter d'autres est possible, mais limité, cette démarche complique la lecture du graphique. J'ai déjà montré dans un précédent article pourquoi une représentation analytique en 3D serait une erreur [4] car elle brouillerait la lisibilité de l'analyse. EAnalysis apporte une solution simple à ce problème : l'utilisation de plusieurs types de vues en simultanée (Figure 1).

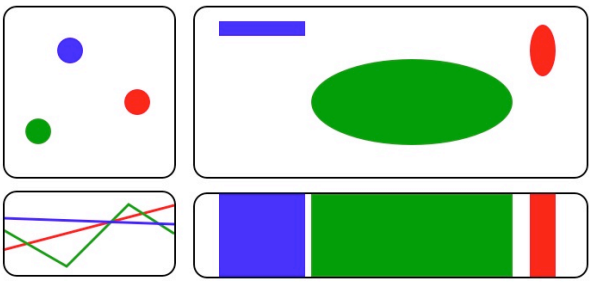


Figure 1. Une représentation combinant plusieurs types de vues.

Ce système de vues multiples permet de visualiser les propriétés d'un *event* à travers des angles différents. J'ai déjà expérimenté avec succès ce type de représentation dans une de mes précédentes analyses [3].

3.2. L'architecture d'EAnalysis

EAnalysis est construit autour de 4 éléments principaux (Figure 2).

3.2.1. Le lecteur audiovisuel

Le lecteur audiovisuel permet d'analyser un ou plusieurs fichiers audio et/ou vidéo dans un même projet. Il est le cœur du logiciel. Ce lecteur contient toutes les fonctions utiles à l'analyste comme la lecture

[8] Les objets graphiques ajoutés sur la représentation se nomment *graphic events* et les objets analytiques *analytic events*.

[9] Par exemple : les positions horizontale et verticale, la forme graphique et la couleur.

en boucle ou la variation de la vitesse de lecture (sans modifier la hauteur). Il est intégré dans la fenêtre principale du projet permettant de naviguer facilement dans les fichiers audio et vidéo et d'éditer la majeure partie des *events*.

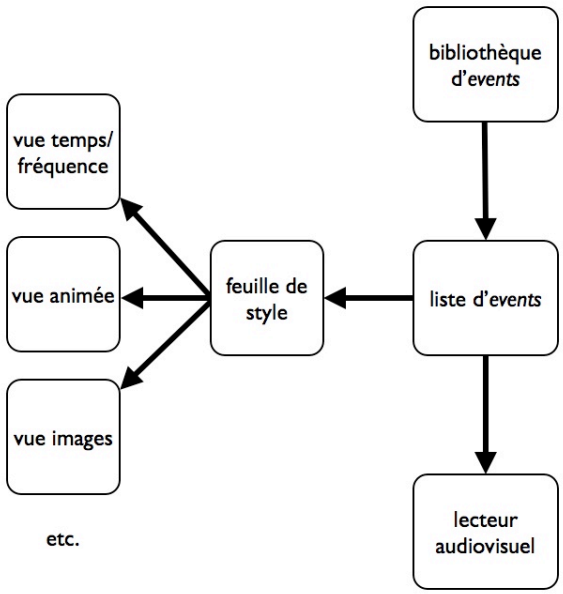


Figure 2. L'architecture d'EAnalysis.

3.2.2. *La bibliothèque d'events*

La bibliothèque d'*events* se divise en deux parties : les *graphic events* et les *analytic events* (Figure 3). Ces deux types d'*events* ne sont en réalité que la présentation différente des mêmes objets. Comme indiqué dans la figure 6, un *event* contient un ensemble de propriétés réparties en trois catégories :

1. les propriétés générales : le nom de l'objet et ses coordonnées temporelles et graphiques ;
2. les propriétés graphiques : le type de forme et toutes les caractéristiques graphiques de l'objet ;
3. les propriétés analytiques : la liste des paramètres d'analyse de l'objet.

Un *graphic event* ne contient pas de paramètre analytique, mais il est possible d'en ajouter tandis qu'un *analytic event* contient la définition de l'ensemble des propriétés liées à l'objet. Ces propriétés peuvent bien évident être complétées ou modifiées par l'utilisateur. Elles se présentent sous la forme d'une liste classée par catégories (sur plusieurs niveaux pour les *analytic events*).

A l'usage, il s'est avéré que la navigation entre les différents *anaytic events* présentés sous la forme de liste n'était pas forcément très pratique ou explicite. C'est la raison pour laquelle j'ai ajouté une fenêtre flottante permettant d'afficher différemment les events (Figure 4). Cette fenêtre contient une image cliquable à partir de laquelle l'utilisateur peut glisser et déposer sur les vues les différents *events*.

3.2.3. *Les vues*

Le troisième élément est un ensemble de vues permettant d'afficher les propriétés graphiques et analytiques des *events*. Ces différentes vues s'affichent sur une seule fenêtre. Dans l'état actuel de EAnalysis, seules les vues temps/fréquences et images[10] ont été développées. Les autres vues seront disponibles au fur et à mesure de l'avancée du logiciel. La figure 5 présente la fenêtre principale avec un exemple de 3 vues horizontales superposées. L'utilisateur peut ajouter autant de vues qu'il le souhaite à cette fenêtre. Les têtes de lecture des différentes vues sont synchronisées, mais les vues peuvent être autonomes dans leur niveau de zoom. Ainsi, dans la figure 5, la vue du bas, représentant la forme d'onde, affiche l'intégralité du fichier audio dans un mode synoptique tandis que les deux vues supérieures sont paramétrées sur un niveau de zoom important afin de montrer les détails. Chacune des vues possède aussi un ensemble de paramètres (fond, couleur, tête de lecture, règle temporelle, masque en fonction d'une autre vue, etc.) qu'il est très simple de modifier. Mais ces vues possèdent aussi un ensemble de paramètres d'affichage des *events* qui seront pris en charge par la feuille de style (voir la section 3.2.4).

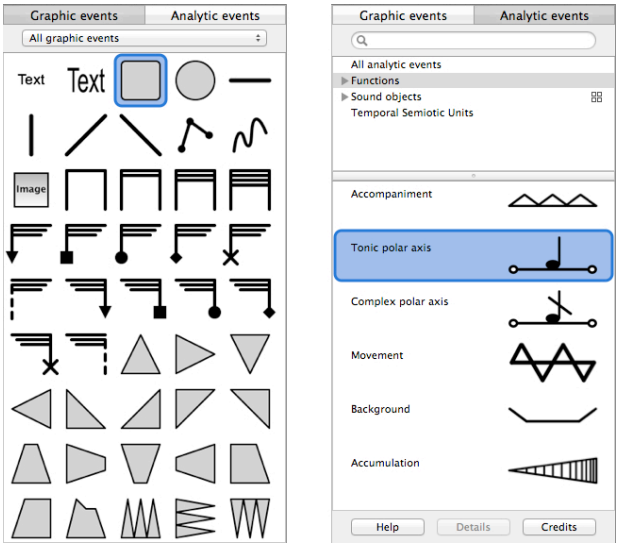


Figure 3. Les deux parties de la bibliothèque d'*events*.

Les vues qui viendront se rajouter à celles-ci permettront de visualiser un *event* ou un groupe d'*events* de différentes manières en créant un focus sur un ou plusieurs paramètres ou sur une structure. Il existe de nombreux modes de représentation analytique de la musique [3], voici les types de vue qui seront ajoutées dans les prochaines mises à jour : une vue animée permettant de représenter les mouvements des sons dans l'espace interne de l'œuvre ou lors de la mise en espace, une vue par *events* permettant de mettre en valeur leurs

[10] La vue image permet de synchroniser une partition musicale ou n'importe quelle suite d'images sur le déroulement temporel. Cette vue est destinée en priorité à l'analyse de la musique mixte.

relations ou afficher une analyse paradigmatique et une vue en graphiques afin d'afficher des données recueillies dans d'autres logiciels.

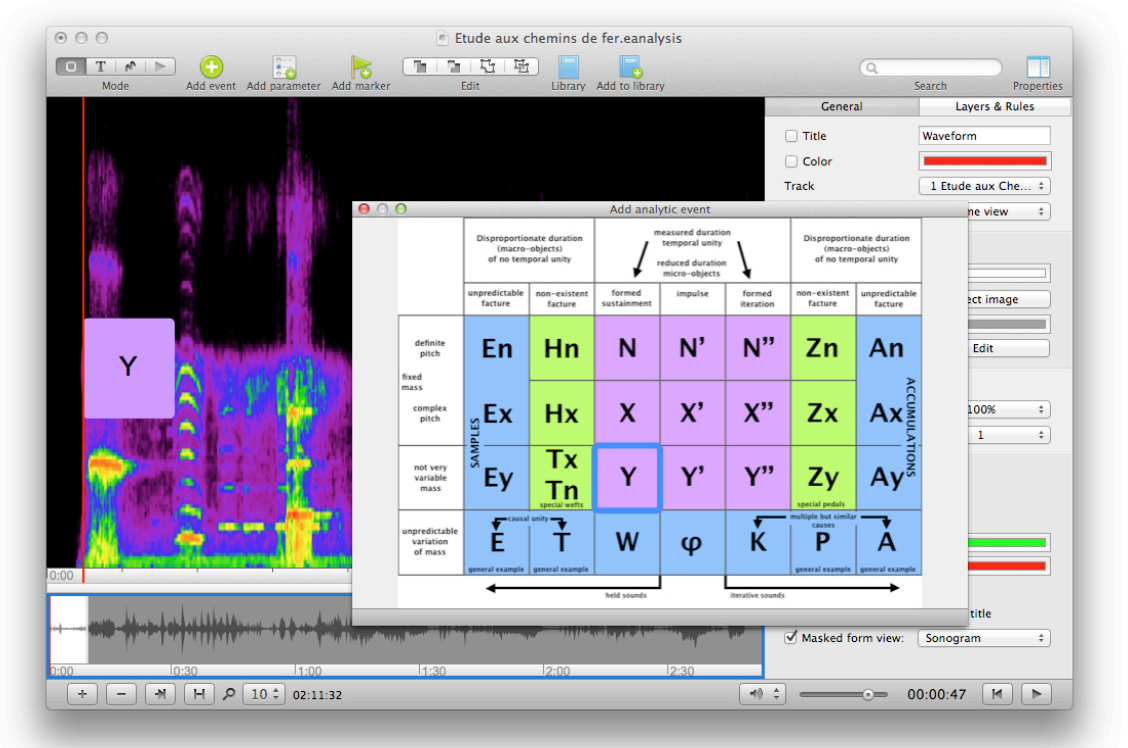


Figure 4. La fenêtre complémentaire de la bibliothèque d'*analytic events* permettant de présenter différemment les *events* : un exemple avec les objets sonores de la typomorphologie de Pierre Schaeffer.

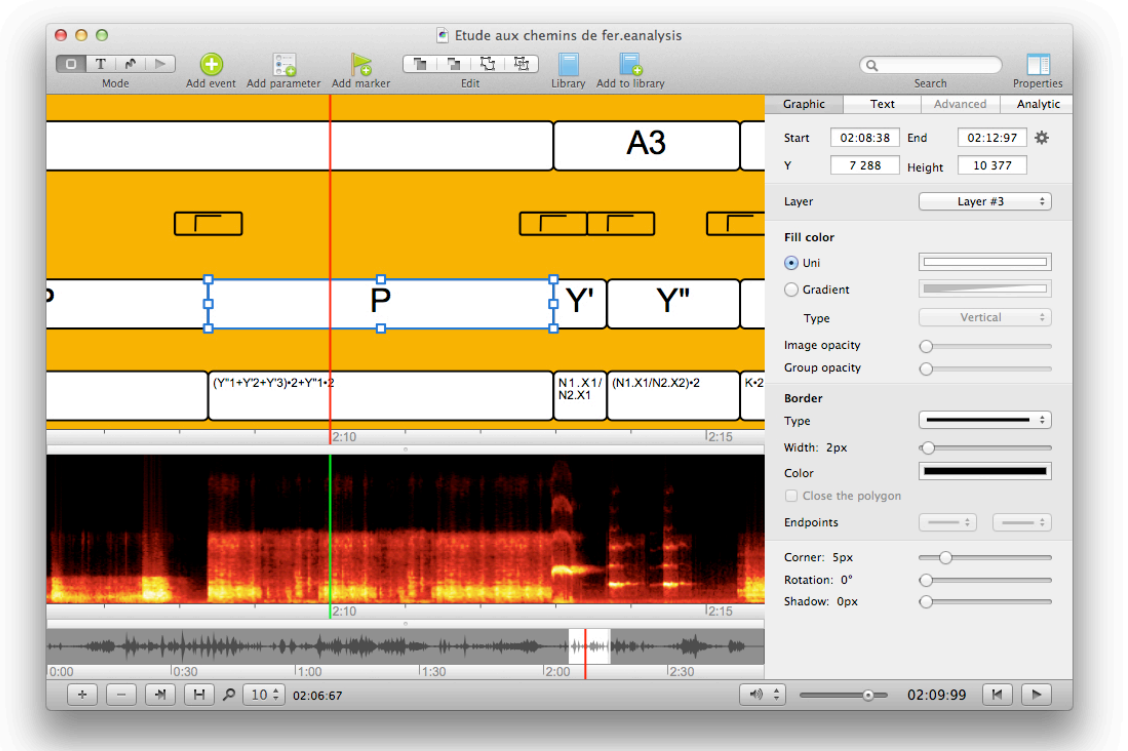


Figure 5. La fenêtre principale avec 3 exemples de vues (dessin, sonagramme, forme d'onde). Ici l'analyse de l'*Etude au chemin de fer* de Pierre Schaeffer par Michael Gatt [11].

3.2.4. La liste d'events et la feuille de style

Le quatrième élément de EAnalysis est le fichier du projet lui-même, c'est-à-dire la liste des *events* associés aux fichiers audiovisuels. Il est complété par une feuille de style permettant de déterminer la manière dont les *events* seront affichés sur les vues (Figure 6). Cette feuille de style n'est pas encore totalement implémentée dans le logiciel et l'utilisateur n'y a pour le moment pas accès. Son fonctionnement est relativement simple : elle reprend un certain nombre de propriétés graphiques et analytiques de l'*event* afin de les convertir ou non en réalité graphique (la manière dont l'*event* sera réellement affiché). Certaines vues, comme la vue temps ou intensités/fréquences, possèdent une feuille de style intégralement paramétrable par l'utilisateur, d'autres vues seront paramétrées avec une feuille de style en partie ou non modifiable. Elle permettra ainsi de modifier très rapidement la manière dont les graphiques s'afficheront. Elle permettra aussi d'afficher ou non des paramètres analytiques afin de tester ou de présenter son analyse. Par exemple, il sera possible de modifier la couleur des *events* en fonction des paramètres analytiques d'espace.

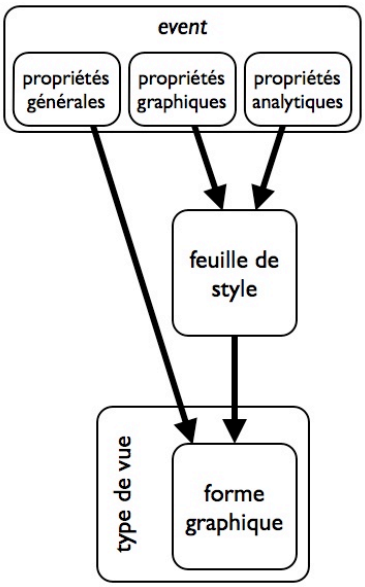


Figure 6. Le système de feuilles de style qui détermine l'affichage des *events* sur les vues.

4. PROCHAINES ETAPES

Comme je l'ai précisé dans l'introduction et rappelé au cours de l'article, le développement d'EAnalysis n'est pas terminé. Entre la version actuelle et la version finale en octobre 2013, de nombreuses fonctions seront ajoutées. Dans cette partie, je vais aborder deux fonctions importantes qui sont en cours de réalisation et seront disponibles dans les mois qui viennent.

4.1. La partie ouverte du logiciel

La bibliothèque d'*analytic events* présentée dans la partie 3.2.2 est conçue pour être modifiable par l'utilisateur. Chaque catégorie principale est enregistrée dans un bundle. Ce bundle contient un ensemble de fichiers en XML et en HTML, d'images en jpeg, de sons en MP3 ou AAC et d'un ensemble de dossiers (Figure 7). EAnalysis sera livré en version finale avec plusieurs bundles, pour le moment, il ne contient que les objets sonores de Pierre Schaeffer, les fonctions de Stéphane Roy et les Unités sémiotiques temporelles développées par le MIM[11] [1].

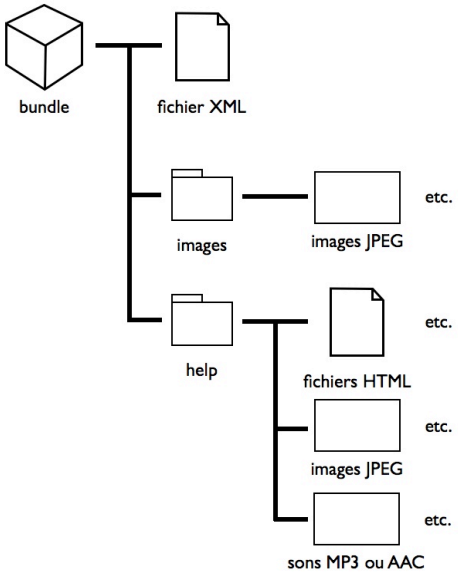


Figure 7. L'architecture des bundles.

En choisissant des formats de fichiers faciles à créer ou éditer (XML, HTML, JPEG, MP3), l'objectif est d'inciter la création d'autres bundles. Toutefois, créer ou modifier un fichier XML (et même HTML) n'est pas à la portée de tout musicologue. C'est la raison pour laquelle, un éditeur de bundle sera ajouté au logiciel avec une interface graphique simple et intuitive. Les chercheurs pourront ainsi créer leurs propres outils d'analyse et les partager avec la communauté.

4.2. Pour qui ? (2/2)

Si l'on revient à la question posée en 2.4, une des idées qui sous-tendent le développement d'EAnalysis est centrée sur l'utilisateur. Quel est le public cible de ce logiciel ? Il semble évident qu'il est très varié : de l'enseignant désirant transmettre un ensemble de savoirs à ses étudiants, au musicologue cherchant à analyser une œuvre, en passant par l'étudiant découvrant la musique électroacoustique ou l'ethnomusicologue analysant ses enregistrements. Cette idée se réalisera dans EAnalysis à travers l'interface. Celle-ci contiendra plusieurs modes d'édition et de navigation :

[11] Laboratoire Musique et Informatique de Marseille.

1. un mode normal dans lequel l'utilisateur analysera l'œuvre à l'aide d'*events* et de paramètres analytiques ;
2. un mode dessin qui permettra de dessiner simplement sur une vue. Ce mode est idéal dans une utilisation avec une tablette graphique ou un tableau blanc interactif. Il permet de noter des idées ou d'exprimer ce que l'on ressent lors d'une écoute en rompant avec le traditionnel déroulement temporel de gauche à droite. Le dessin reste synoptique ou peut s'animer en fonction de la lecture audiovisuelle ;
3. un mode texte qui permettra à l'utilisateur de noter ses idées durant la lecture. Le résultat se traduira par des *text events* affichables aussi sous la forme d'une liste ;
4. un mode lecture dans lequel l'ensemble des *events* seront verrouillés afin de permettre à l'utilisateur de naviguer dans le document sans risquer de le modifier et d'activer les propriétés de liens[12].

Enfin, EAnalysis offrira aussi la possibilité d'importer ou d'exporter les données d'autres logiciels tels que, par exemple, iAnalyse, l'Acousmographe, Audiosculpt[13] ou Sonic Visualiser[14] à travers les formats texte ou XML. Une intégration OSC est aussi prévue afin de connecter Eanalysis à des environnements tels que Max.

5. CONCLUSION

Cet article présente le projet de développement du logiciel EAnalysis proposant des fonctions innovantes pour l'analyse de la musique électroacoustique. Le projet est en cours de réalisation, de nombreuses fonctions manquent encore et seront ajoutées au fil des mois qui viennent.

On pourrait regretter l'absence de modules d'analyse automatique ou semi-automatique qui permettraient par exemple d'aider l'analyste dans la segmentation du matériau [12] ou dans l'analyse par descripteurs [16] mais l'objectif de ce projet n'est pas de réaliser le logiciel idéal, il s'agit plutôt d'expérimenter de nouveaux modes de représentations et d'analyses. De plus, les fonctions d'importation décrites précédemment permettront d'intégrer des données recueillies dans d'autres logiciels.

Enfin j'espère qu'EAnalysis n'est qu'un début. Ces nouvelles pratiques et méthodes ont besoin d'être affinées, améliorées et enrichies par d'autres chercheurs qui développent ou développeront d'autres logiciels pour l'aide à l'analyse de la musique électroacoustique.

[12] Chaque *event* pourra contenir un lien vers une position temporelle du même projet ou d'un autre projet, vers un fichier ou vers une page web.

[13] Audiosculpt est développé par l'Ircam (Paris) : http://forumnet.ircam.fr/691.html.

[14] Sonic Visualiser est développé par le *Centre for Digital Music* de l'université Queen Mary de Londres : http://www.sonicvisualiser.org.

6. REFERENCES

[1] Coll. *Les unités sémiotiques temporelles*, MIM, Marseille, 1996.

[2] Couprie, P. "Analyse de Jukurrpa – Quatre rêves", Musimédiane, n°1, Paris, 2005.

[3] Couprie, P. "(Re)Presenting electroacoustic music", Organised Sound, vol. 11 n°2, Cambridge, 2006.

[4] Couprie, P. "Dessin 3D et système immersif pour la représentation de la musique électroacoustique", Electroacoustic Music Studies Network, Leicester, Grande Bretagne, 2007.

[5] Couprie, P. "iAnalyse : un logiciel d'aide à l'analyse musicale", Journées d'Informatique Musicale, Albi, France, 2008.

[6] Couprie, P. "La représentation graphique : un outil d'analyse et de publication de la musique électroacoustique", Doce Notas, L'analyse de la musique, n° 19-20, Madrid, 2009.

[7] Couprie, P. "Utilisations avancés du logiciel iAnalyse pour l'analyse musicale", Journées d'Informatique Musicale, Rennes, France, 2010.

[8] Desainte-Catherine, M. Di Santo, JL. "L'acousmoscribe, un éditeur de partitions acousmatique", Electroacoustic Music Studies Network, Buenos Aires, Argentine, 2009.

[9] Emmerson, S. "The relation of language to materials", The language of electroacoustic music, Basingstoke, 1986.

[10] Favreau, E. Geslin, Y. Lefèvre, A. "L'acousmographe 3", Journées d'Informatique Musicale, Rennes, France, 2010.

[11] Gatt, M. "Michael Gatt's Etude aux chemins de fer analysis", OREMA Project, Leicester, 2011, http://www.orema.dmu.ac.uk/?q=content/michael-gatt's-etude-aux-chemins-de-fer-analysis.

[12] Gulluni, S. Buisson, O. Essid S. Richard G. "An interactive system for electro-acoustic music analysis", ISMIR, Miami, USA, 2011.

[13] Landy, L. *Understanding the Art of Sound organization*. MIT Press, Cambridge, 2007.

[14] Roy, S. *L'analyse des musique électroacoustiques : modèles et propositions*, L'Harmattan, Paris, 2003.

[15] Schaeffer, P. *Traité des objets musicaux*, Le Seuil, Paris, 1966.

[16] Schwarz, D. "Principles and applications of interactive corpus-based concatenative synthesis", Journées d'Informatique Musicale, Albi, France, 2008.

[17] Smalley, D. "Spectromorphology : explaining sound-shapes", Organised Sound, vol. 2 n°2, Cambridge, 2001.

L'ETHNOMUSICOLOGIE ET L'INFORMATIQUE MUSICALE : UNE RENCONTRE NECESSAIRE

Stéphanie Weisser
Musée des Instruments de Musique
s.weisser@mim.be

RÉSUMÉ

L'ethnomusicologie – en tant que discipline qui étudie les musiques qui n'appartiennent pas à la tradition occidentale savante et leurs modalités d'articulation avec la société dans laquelle elles existent – entretient avec l'informatique musicale des rapports étroits. De la collecte des données audiovisuelles à l'analyse de ces dernières, des outils informatiques sont utilisés à toutes les étapes de la recherche. Pourtant, les ethnomusicologues ne disposent pas, à ce stade, d'un outil informatique intégré spécifiquement pensé et conçu pour leur discipline.

En outre, les musiques dont l'étude relève de l'ethnomusicologie sont de plus en plus souvent produites à l'aide de dispositifs informatiques, qui ont un impact sur les catégorisations et conceptualisations sonores et musicales des cultures qui les ont intégrés dans leur pratique musicale. Pourtant, l'analyse approfondie de ces dispositifs et de leurs effets est rarement effectuée, par manque d'outils méthodologiques aussi bien que par l'écartement de l'objet sonore ainsi produit du champ de la discipline.

A la fois outil de l'étude et objet d'étude, l'informatique musicale nécessite donc une réflexion épistémologique et méthodologique par rapport à son utilisation dans le domaine ethnomusicologique.

1. INTRODUCTION

L'ethnomusicologie est une discipline dont la naissance et le développement sont fondamentalement liés aux possibilités techniques d'enregistrement et d'analyse du son. Historiquement, on peut même situer sa naissance en 1886, date de la parution de l'article d'Alexander J. Ellis, « On the Musical Scales of Various Nations ». La généralisation de l'utilisation du phonographe, puis d'autres moyens d'enregistrement (notamment vidéo) peut être considérée comme une évolution technique qui a eu un impact très important sur les fondements conceptuels et sur les pratiques méthodologiques de la discipline [5].

En effet, l'enregistrement permet la réécoute, et l'analyste n'est plus limité/e à la capacité de sa mémoire immédiate pour observer les caractéristiques formelles du répertoire qu'il/elle écoute. Cette « fixation » a aussi permis de modifier à volonté l'échelle temporelle de l'événement musical : l'enregistrement donne en effet accès aux événements sonores aussi bien dans leur dimension temporelle globale qu'à des détails de très courte durée, et ce aussi souvent que souhaité. Enfin, en fonction des méthodes employées lors de la prise de son, il est également devenu possible de recueillir séparément les différentes parties d'une polyphonie et d'accéder ainsi aux parties constitutives de cette dernière.

La numérisation de l'enregistrement musical (soit lors de la prise de son soit *a posteriori*) a permis de résoudre un certain nombre de problèmes, comme ceux liés à la manipulation et à la dégradation des supports [6] ou à la reproductibilité – tout en créant de nouveaux (capacités de stockage et gestion des métadonnées, notamment). Néanmoins, cette numérisation constitue un atout très important pour la discipline : tout d'abord, elle permet dorénavant d'intégrer – dans une certaine mesure – une dimension historique à court terme. Elle facilite également la duplication et la dissémination des sources, et notamment la préservation de celles-ci au sein des sociétés étudiées, qui ont ainsi accès à leur propre passé musical. Enfin, elle facilite la réalisation, sur le terrain même, d'expérimentations interactives, ou simplement de réécoutes commentées par les musiciens eux-mêmes. La captation sonore est elle aussi fortement simplifiée : il suffit pour cela de comparer les méthodes d'enregistrement en parties séparées mises en point par Simha Arom dans les années 1970 [2] avec l'enregistrement multipiste des années 2000 [25].

La numérisation de l'enregistrement musical a aussi permis l'utilisation et l'élaboration de nombreux outils, traitements et analyses informatiques. Parmi ces outils, de nombreux logiciels ont été développés dans le domaine de l'édition sonore (montage), de l'aide à l'analyse musicale, de l'extraction d'informations sonores (calculs de descripteurs de haut et bas niveau) et de l'édition de représentations graphiques (partitions). L'ethnomusicologue, en fonction des sujets étudiés, fait régulièrement appel à tous ou certains de ces types de logiciels.

Parallèlement, les objets d'étude de l'ethnomusicologie ont également été modifiés par la démocratisation et la généralisation des outils informatiques appliqués à la musique. Depuis la prise de

son jusqu'à l'utilisation d'effets proposés par les logiciels de montage, de l'emploi de sons de synthèse (instruments virtuels) à l'adoption de structures formelles proposées « par défaut » (puisque commerciales), les exemples d'adoption des technologies numériques dans les musiques non occidentales sont légion. Or, ce type de musiques, souvent désignées par les termes de « populaires », « néotraditionnelles » [23], « métissées », « modernes » etc. ont souvent été exclues du champ d'investigation des chercheurs formés à l'ethnomusicologie de l'urgence – comme si l'intégration de moyens technologiques, certes d'origine occidentale, constituait une « contamination » telle que le répertoire concerné en devenait non pertinent par rapport au champ de la discipline .

La naissance, le développement et la généralisation d'outils informatiques appliqués à la musique ont donc un impact important sur l'ethnomusicologie, tant en termes de possibilités méthodologiques (outils d'analyse) que de positionnement épistémologique (définition de son objet). Il est donc important de mener ce constat en questionnant certains présupposés par rapport aux pratiques musicales et ethnomusicologiques actuelles. Ce papier se propose donc d'adresser certaines de ces questions, en se fondant sur des observations et réflexions menées dans le cadre d'une pratique de recherche et d'enseignement en ethnomusicologie.

2. L'INFORMATIQUE COMME OUTIL POUR L'ETHNOMUSICOLOGIE

L'application d'outils informatiques à l'analyse ethnomusicologique n'est pas neuve. Dès les années 1960, des ethnomusicologues ont cherché à utilisé l'ordinateur pour organiser les données musicales (préalablement analysées et encodées) et les soumettre à divers traitement statistiques [31], [32]. Plus récemment, Simha Arom et son équipe ont imaginé et réalisé des outils et des dispositifs interactifs pour l'analyse et l'expérimentation [3]. De l'étude des échelles des xylophones africains étendue aux métallophones javanais [37] puis à celles d'autres instruments et sources sonores comme les voix chantées (en contexte polyphonique) des pygmées Bezdann et les flûtes des Ouldémé (Cameroun) [25], la démarche initiée par Simha Arom a certainement été l'une des plus fructueuses, tant d'un point de vue ethnomusicologique (par le développement de concepts, de grilles d'analyse et de méthodologies pionnières) que d'un point de vue technologique, puisqu'il a fallu inventer et développer des outils nouveaux afin de répondre aux problèmes posés, comme par exemple le refus des musiciens africains de travailler avec des sons « clairs », ou d'offrir aux musiciens la possibilité d'expérimenter une solution proposée via un geste musical connu et habituel.

Actuellement, les ethnomusicologues utilisent fréquemment des outils informatiques dans leurs pratiques de recherche, et pour répondre à des besoins variés. L'étude des échelles est probablement l'un des sujets d'étude pour laquelle l'utilisation d'outils informatiques est devenue la plus répandue (voir notamment [29]). Elle repose le plus souvent sur des analyses de hauteurs des sons (en Hz) et des calculs d'intervalles (en cents).

Le domaine du timbre commence également à être investigué par les ethnomusicologues, qui ont analysé (à l'aide d'outils informatiques) le timbre des *sanza* africaines [14], le rôle de la rugosité auditive dans différents répertoires [35], une possible « typologie » des timbres [13] ainsi que des différents mécanismes et caractéristiques de répertoires vocaux, comme le chant diphonique et la *quintina* de Sardaigne (brillamment visualisées, ainsi que d'autres analyses ethnomusicologiques, via une animation interactive en ligne, les *Clés d'écoute* [1]).

L'analyse des rythmes s'appuie fréquemment sur l'utilisation d'outils informatiques, mais l'apport de ces derniers consiste souvent en l'appui à la segmentation du flux musical (identification de l'attaque et du début d'un son) dans le but de parvenir à une mesure précise des durées (voir par exemple [30], [15]). Quant à l'analyse informatisée du geste musical, même si des solutions récentes permettent une collecte des données sur le terrain, en contexte, elle nécessite néanmoins encore une technicité assez importante, qui peut s'avérer problématique.

Enfin, les ethnomusicologues utilisent largement des logiciels de montage audio et/ou vidéo, ainsi que des logiciels d'analyse acoustique avec interface graphique (*Audiosculpt*, *Spear*, *Praat* [10]). Les outils de calcul (tableurs) figurent eux aussi dans la « boîte à outils » des praticiens de la discipline (voir notamment le logiciel de description et d'aide à l'analyse des monodies intitulé *Monika* [27] et constitué d'une macro pour *Excel*).

Des éditeurs de partition sont aussi utilisés très régulièrement. Il est d'ailleurs intéressant de noter que ces logiciels peuvent également être utilisés pour leur fonctionnalité de jeu des partitions encodées, surtout lorsque l'utilisation d'un son défini par l'utilisateur est possible. Cette fonctionnalité permet de soumettre au jugement des musiciens une proposition musicale, et ainsi de tester/affiner l'hypothèse ayant présidé à la formulation de cette proposition [38].

Ce type d'expérimentation a été mené de manière bien plus sophistiquée et systématisée dans le cadre d'une analyse sur les échelles à l'aide de patches spécifiquement développés pour *OpenMusic*, comme *Scala* [25], *PARETO* [21]) ou d'applications développées sous *Matlab* [18]. Ces travaux, souvent

menés conjointement par des chercheurs de disciplines différentes, démontrent la richesse et l'intérêt de ce type de recherches « croisées ».

Par contre, les outils d'aide à l'analyse musicale semblent être nettement moins fréquemment utilisés en ethnomusicologie. Si les dispositifs de transcription automatique avaient soulevé beaucoup d'espoir lors de leur création (notamment en termes d'objectivité, d'infaillibilité et de précision [19]), leur usage ne semble s'être guère généralisé. Les outils actuels disponibles[1] ne semblent pas non plus avoir convaincu les utilisateurs ethnomusicologues, qui ne se sont emparés ni des logiciels dont la conception s'inspire des « annotateurs » (qu'ils dépassent cependant très largement) comme *iAnalyse* ou l'*Acousmographe,* ni de logiciels « computationnels » comme *Sonic Visualizer* [7], ou *Humdrum* [11] – sauf parfois lors de collaborations comme celles évoquées ci-dessus. Serait-ce par inadéquation des outils, par méconnaissance des potentialités offertes ou par manque de courage devant la complexité technique, réelle ou supposée, de ces logiciels ?

Pourtant, on a assisté ces dernières années à la naissance d'une nouvelle sous-discipline potentiellement très intéressante : l'ethnomusicologie computationnelle [34]. Cette dernière se définit comme « la création, le développement et l'utilisation d'outils informatiques potentiellement utiles à la recherche ethnomusicologique » [*Ibid.*, p. 1]. Outre qu'on pourrait trouver l'expression un peu étonnante (il semblerait en effet plus logique de parler *d'informatique appliquée à l'ethnomusicologie*, puisque l'objectif de la recherche est bel et bien la production d'un outil informatique) sont rassemblés sous cette « bannière » des outils, des méthodes et des objectifs très divers, parmi lesquels se trouvent des outils d'aide à la représentation graphique et des logiciels d'analyse et de mesure de paramètres (psycho-)acoustiques (tels que la hauteur des sons et de rythme). Or, ces outils d'analyse *a posteriori* procèdent de la démarche analytique habituelle, qui cherche à extraire les informations nécessaires d'un événement musical déjà fixé, enregistré et sont, comme exposé ci-dessus, régulièrement utilisés par des ethnomusicologues « classiques ». Ces derniers feraient-ils, à l'instar de Monsieur Jourdain, de l'ethnomusicologie computationnelle sans le savoir ? Ou suffirait-il d'utiliser un ordinateur dans sa recherche pour faire de l'ethnomusicologie computationnelle ?

Les autres types d'outils et démarches évoqués proviennent du champ de la *Music Information Retrieval* et consistent plutôt en l'application de techniques d'extraction automatique d'information, de traitement numérique du signal ou d'analyse du mouvement (souvent développés dans le cadre de répertoires occidentaux – classiques ou commerciaux) à des musiques dites « traditionnelles » ou « ethniques». Or, cette démarche ne revient pas pour autant à faire de l'ethnomusicologie ! En effet, l'intégration de la dimension *etic*, de la pertinence culturelle et de la prise en compte de l'altérité des systèmes et des concepts musicaux et sonores, qui constituent des fondamentaux en ethnomusicologie, est souvent minimale.

Enfin, une autre catégorie référencée comme relevant de l'ethnomusicologie computationnelle regroupe des approches qui font partie prenante de la démarche de réflexion, dès la collecte d'information. Autrement dit, elles sont véritablement constitutives de la démarche de recherche. Font partie de cette catégorie les dispositifs déjà évoqués qui font appel à des expérimentations perceptives interactives menées sur le terrain (voir ci-dessus), à l'apprentissage automatique (*machine learning*) ou à une mise en place dédiée (en vue de l'analyse du geste par exemple (voir les travaux de Martin Clayton, comme par exemple [8], et [9]). Dans ce cas, et dans ce cas seulement, il devient possible, à mon sens, de parler d'ethnomusicologie computationnelle, d'autant que cette démarche n'est possible que lorsqu'une analyse plus « conventionnelle » préparatoire et fouillée a été réalisée.

Quels sont les outils informatiques nécessaires à l'analyse ethnomusicologique – computationnelle ou conventionnelle ? L'ethnomusicologie au sens développé par Simha Arom a pour objectif général de comprendre et de formaliser les règles souvent implicites qui président aux performances musicales dans un système donné. Pour ce faire, l'ethnomusicologue collecte un corpus cohérent de performances, l'analyse (à l'aide de différentes méthodes) et cherche à formuler les « règles » qui expliquent l'organisation interne des pièces analysées et, s'il le peut, teste ces hypothèses auprès des musiciens qui les valident ou les invalident. La difficulté de la démarche est précisément là : comme les règles ne sont pas explicitées (ou pas complètement), il est peu aisé d'évaluer à l'avance si tel outil, telle méthode d'analyse sera utile, pertinente et fructueuse. Si ce type d'incertitude n'est pas forcément spécifique à l'ethnomusicologie – les musiques « contemporaines », par exemple, peuvent également être caractérisées par cette absence de standardisation formelle – la difficulté est peut-être amplifiée par l'altérité fondamentale des systèmes de pensée entre le producteur et l'analyste.

Néanmoins, les travaux fondateurs dans la discipline permettent de dégager des éléments dont l'analyse est importante, de manière générale [4]:

1. Les hauteurs ou intervalles entre les hauteurs des sons et leur hiérarchisation éventuelle (se traduisant

[1] Voir notamment une recension de ces outils (essentiellement computationnels) sur la page http://smcnetwork.org/view/software

parfois par leur prégnance plus ou moins importante dans l'événement musical, mais pas toujours),

2. L'organisation temporelle, qui s'analyse notamment par l'examen des relations (proportionnelles ou non) entre les durées des sons, de la présence éventuelle d'une pulsation (étalon isochrone), matérialisée ou non, de la combinaison éventuelles des pulsations en cycles et de la différentiation entre temps forts et temps faibles

3. La récurrence d'événements musicaux (courts ou longs) et leurs éventuelles variations,

4. L'organisation éventuelle en différentes parties simultanées (polyphonie) et la manière dont ces parties simultanées sont organisées les unes par rapport aux autres et par rapport à l'ensemble

5. Les caractéristiques des timbres des sons musicaux (instrumentaux et/ou vocaux) et leurs éventuelles variations.

L'ethnomusicologue a également besoin d'un outil graphique (le plus souvent un éditeur de partition, mais pas toujours) qui lui permette de transcrire sous forme écrite un élément sonore qu'il cherche à modéliser. Il faut d'ailleurs souligner que cette opération de transcription est une des premières phases de l'analyse structurelle d'une pièce musicale ; dans la pratique, ces deux tâches s'opèrent simultanément, en parallèle. Cette articulation, ce va-et-vient entre transcription et analyse est peut-être, avec la prise en compte de la dimension culturelle de la musique (qui influe très profondément la manière de *faire* et de *penser* cette musique), une des spécificités fondamentales de la *praxis* de l'ethnomusicologue, si compliquée à définir.

Il est important de préciser que la grille d'analyse présentée ci-dessus n'est ni exhaustive ni, probablement, valable pour tous les répertoires étudiés ou à étudier par l'ethnomusicologie. Ainsi, plusieurs travaux ont montré que les hauteurs, même relatives, sont parfois considérées comme moins importantes que les contours de la mélodie [33] ou aussi importantes que la couleur sonore [36]. De même, une performance classique de l'Inde du Nord, fondée sur un système musical d'une grande sophistication (tant au niveau de l'organisation des hauteurs sonores que des durées) et largement théorisée ne nécessitera pas les mêmes outils (ou les mêmes configurations d'outils) qu'une pièce polyphonique de trompes centrafricaines fondées sur une organisation en hoquet dont les règles d'exécution et de variation sont apprises par imprégnation. Et aucun de ces systèmes n'est réductible au système tonal occidental « classique », qui, déjà, avait montré son insuffisance dès lors que l'objet d'analyse déborde d'un groupe très limité d'œuvres produites dans un cadre historiquement, géographiquement et sociologiquement circonscrit.

Or, la plupart des outils informatiques destinés à l'analyse de la musique sont fondés sur des présupposés provenant de ce système musical tonal. Ceci est particulièrement vrai pour les éditeurs de partitions disponibles sur le marché, qui, malgré des efforts louables, peinent à intégrer les spécificités propres aux musiques « autres » (ce qui inclut les musiques contemporaines, les musiques populaires, les musiques électroniques, etc.).

Les logiciels « computationnels » qui extraient des informations à partir des fichiers sonores font face à des difficultés similaires. Par exemple, les analyses de tempo proposées par *Sonic Visualizer* [7] et *MIRToolbox* [20] – deux outils absolument remarquables – sont implicitement fondées sur l'idée que la musique à analyser est composée de durée proportionnelles les unes aux autres et fondée temporellement sur une pulsation. Le problème n'est certes pas dû aux concepteurs de ces outils : ils fondent leur réflexion sur les travaux relatifs à la perception de la musique. Or, ces deniers sont menés la plupart du temps avec des auditeurs habitués au système musical occidental et avec des stimuli produits dans le cadre de ce système. Les conclusions de ces travaux ont peut-être une portée plus large l'univers sonore tonal et mesuré, mais peut-être pas.

En résumé, un outil informatique véritablement propre à l'ethnomusicologie permettrait d'être configuré facilement en fonction des situations rencontrées et permettrait de choisir les types d'analyse et de représentation qui seraient souhaitées et utiles. Par exemple, la possibilité de visualiser simultanément plusieurs éléments synchronisés (à la manière d'*iAnalyse* [12]) serait certes des plus utiles. Une articulation aisée entre des analyses et/ou computations à un niveau détaillé (portant sur des sons isolés ou en nombre limité) et plus globale (totalité d'une pièce) serait également précieuse. L'intégration de la dimension visuelle (vidéo) permettrait de mettre en rapport des éléments relatifs au sonore et au gestuel (producteur du son instrumental ou danse, par exemple). Un module de d'encodage en notation occidentale très flexible serait apprécié. Enfin, l'intégration de la possibilité de générer des propositions musicales et d'encoder les évaluations de celles-ci par les musiciens compléterait utilement les possibilités d'un tel logiciel. Il est plus que probable que de nombreux autres souhaits, peut-être même contradictoires, pourraient être formulés en fonction des spécificités des musiques rencontrées.

Par contre, un élément important quel que soit le sujet d'étude serait la simplicité d'utilisation de l'interface : l'apprentissage des programmes informatiques tels que *Matlab*, par exemple, est long et peu aisément réalisable de manière autonome ou autodidacte – ou en tout cas sans avoir bénéficié d'une

formation approfondie en informatique et/ou en programmation, ce qui est en général le cas de l'utilisateur/trice ethnomusicologue moyen/ne.

3. L'INFORMATIQUE COMME OBJET D'ETUDE POUR L'ETHNOMUSICOLOGIE

Comme l'ont souligné plusieurs auteurs [16], [22], les musiques produites avec l'aide d'outils informatiques font maintenant partie du paysage musical mondial. Cette irruption de la technologie dans le champ d'événements sonores jusqu'alors purement acoustiques n'est pas nouvelle dans les musiques appartenant au champ socio-culturel et géographique étudié par l'ethnomusicologie. Déjà, la généralisation de l'amplification, de la radio, puis du lecteur de cassettes audio (allant souvent de pair avec l'installation de studios d'enregistrement et donc d'une production musicale locale), puis de synthétiseurs (cf. figure 1) ont déjà en leurs temps été considérées comme potentiellement néfastes à l'authenticité des musiques « traditionnelles » [22, p. 5-6]. La présence de la technologie est donc, depuis plus d'un demi-siècle, vue par certains chercheurs [24] comme une porte ouverte vers une forme de colonisation culturelle du système musical occidental sur les autres formes sonores dans le monde.

Figure 1. Musicien jouant d'un synthétiseur amplifié lors d'une procession à Calcutta (Inde). Photo : Stéphanie Weisser, 14 avril 2011.

Certes, l'expansion de l'utilisation de la technologie et de l'informatique musicale s'opère dans un contexte socio-économique précis : celui d'une industrialisation d'un marché musical, prioritairement urbain et, consécutivement, de l'émergence d'une culture de masse. Les cas sont nombreux [23]. Ils s'accompagnent en général d'une reconfiguration du statut des musiciens, des habitudes d'écoute, de consommation et des formes musicales. En effet, comme l'a souligné Denis-Constant Martin [26], il ne s'agit pas simplement d'un « plaquage » pur et simple d'éléments du langage musical occidental sur une culture musicale existante : des formes nouvelles sont créées, qui correspondent à une évocation, parfois symbolique mais aussi parfois plus littérale, des changements sociaux, politiques ou environnementaux. A ce titre, l'ethnomusicologie s'intéresse légitimement (quoique plutôt rarement) à ces pratiques, qui sont significatives : elles peuvent fournir des informations précieuses sur la manière dont la tradition, la modernité, le « soi », « l'autre » et le statut du musicien (pour ne citer que quelques exemples) sont conçus aujourd'hui dans ces sociétés.

Figure 2. Jaquette d'une cassette audio de chants spirituels de l'église orthodoxe d'Ethiopie, produite et vendue par un magasin spécialisé d'Addis Abeba, Ethiopie, 2005.

Au niveau socio-économique, la perte financière que constitue le remplacement des musiciens par des instruments virtuels est souvent dénoncée par les musiciens. Cette « virtualisation » concerne en priorité les instruments rythmiques et/ou fortement répétitifs, dont l'apport est plus facilement automatisable.

Cependant, avec l'amélioration de la qualité et de la facilité de manipulation des instruments virtuels disponibles, il est réaliste d'envisager une dépossession plus générale des instrumentistes. En effet, dans de nombreuses cultures, l'instrumentiste est également le compositeur ou l'arrangeur de la pièce ou du moins de sa contribution à l'ensemble ; si le jeu de l'instrument devient informatisé, l'acte de création musicale tout entier risque d'être déplacé vers la personne qui manipule l'ordinateur (producteur/ingénieur du son/propriétaire du matériel). Dans certains contextes comme le Sénégal, une dépossession similaire des musiciens au profit des producteurs/vendeurs a déjà été observée [28, p. 56-57]; la généralisation de la MAO, dans ce contexte, risque donc d'accélérer une tendance déjà en marche.

En outre, si, dans certains contextes, la démocratisation du matériel (hardware et software), ainsi que des supports de diffusion (CD) et de

duplication, permet aux musiciens de s'équiper eux-mêmes (sans passer par des studios) et de s'auto-produire, cette facilité de production de copies de bonne qualité fait empirer une situation existante déjà peu favorable aux musiciens, due à l'absence ou aux violations systématiques de réglementations protectrices de leurs droits sur leurs enregistrements (copyright).

L'informatisation globale offre aux musiciens un espace de diffusion et de communication extraordinaire : les plates-formes audiovisuelles telles que Youtube permettent aux musiciens d'accéder à une visibilité internationale – et de décrocher ainsi des engagements en Europe et/ou aux Etats-Unis, générateurs d'une amélioration parfois considérable de leur statut social et financier.

La manière pratique dont les outils de la MAO sont utilisés peut également fournir des informations sur la manière de penser, de percevoir et de travailler le son musical au sein d'une culture. La technologie au sens large et l'informatique en particulier sont des outils du faiseur de musique, au même titre qu'un instrument. Si les banques de sons pré-enregistrés fournies avec les logiciels de montage sont essentiellement constituées d'instruments occidentaux – et constituent à ce titre un risque probable de standardisation et d'appauvrissement des palettes sonores traditionnelles, il faut néanmoins souligner que les éditeurs commencent à proposer des sons issus d'autres traditions musicales ainsi que des possibilités de les agencer selon un système (d'échelles notamment) différent du langage tonal occidental.

En outre, la présence d'effets sonores peut être associée symboliquement à des contenus spécifiques et dépendants à la fois du contexte musical et de la culture. La *reverb* appliquée à l'introduction parlée (et parfois à l'intégralité) des chants spirituels *mezmour* d'Ethiopie (cf. figure 2) a pour objectif de renforcer la solennité et l'importance symbolique et religieuse des paroles prononcées. Par contre, au Népal, l'adjonction d'un léger effet de *reverb* est généralement considérée comme apportant de la douceur à la voix chantée [17].

4. CONCLUSION

L'ethnomusicologue est confronté/e en permanence à l'informatique, dans sa propre démarche scientifique comme dans l'objet qu'il/elle étudie. Il est donc important, à ce titre, qu'il/elle mène une réflexion critique sur les enjeux tant techniques que théoriques de l'informatique musicale.

Souvent, l'ethnomusicologue analyse les éléments extra-musicaux (valeurs, associations sémantiques et symboliques, etc.) liés à l'utilisation d'un instrument de musique dans le contexte étudié. Or, en tant qu'élément participant à la génération d'événements sonores, l'outil informatique devrait être considéré comme un véritable « instrument de la musique » : ses spécificités d'utilisation, son apprentissage, peut-être même ses « techniques de jeu » et surtout les modalités de son emploi en vue de parvenir à un résultat sonore et musical signifiant pourraient être investiguées avec fruit – sans oublier ce dernier !

L'ethnomusicologie doit également mener une réflexion sur la place de l'informatique dans sa propre pratique de recherche. En effet, elle utilise des outils développés pour d'autres disciplines : la linguistique, l'acoustique, l'analyse de la musique occidentale, la psychologie expérimentale et les sciences cognitives, la composition ou les performances musicales en temps réel. La diffusion des résultats d'analyses nécessite aussi une visualisation (soit via un éditeur de partitions ou autre interface graphique pour les transcriptions, soit via une représentation interactive), là aussi réalisée sans outils spécifiques.

Si ces « emprunts » ont certainement encouragé les ethnomusicologues à se montrer inventifs, il est probable que ces derniers seraient particulièrement enthousiastes à l'idée de disposer d'un outil intégré, autonome, permettant de mener au sein d'un environnement unique les différentes étapes de leur travail, depuis l'acquisition et le montage sonore jusqu'à la présentation des résultats, en passant par les analyses variées à des niveaux divers et les expérimentations (éventuellement interactives). Et pas seulement parce qu'un tel outil leur faciliterait considérablement la tâche : la nécessaire flexibilité, couplée à la variété des tâches à effectuer par un tel environnement nécessiterait certainement une réflexion importante, multidisciplinaire et à portée universaliste, puisqu'elle aurait pour objectif de créer un méta-outil propre à analyser *toutes* les musiques. Là pourrait se trouver le vrai défi de l'ethnomusicologie computationnelle.

5. REFERENCES

[1] Armani, A., Chemillier, M., Lortat-Jacob, B., & Rappoport, D., *Clés d'écoute*. Animations musicales interactives en ligne. http://ehess.modelisationsavoirs.fr/ethnomus/ (avec la collaboration de Michèle Castellengo et Gilles Léothaud pour la *quintina* de Sardaigne et de Tran Quang Hai pour le chant diphonique)

[2] Arom, S. "The Use of Play-Back Techniques in the Study of Oral Polyphonies", *Ethnomusicology* 20/3, 1976, p. 483-519.

[3] Arom, S. "L'étude des échelles dans les musiques traditionnelles : une approche interactive", *Analyse Musicale* 23, 1991, 21-24.

[4] Arom, S, *La boîte à outils d'un ethnomusicologue*, Presses de l'Université de Montréal, Montréal, 2007.

[5] Boilès, C. & Nattiez, J.-J. "Petite histoire critique de l'ethnomusicologie", *Musiques en jeu* 28, 1977, p. 26-53.

[6] Brice, G. "Ethnomusicologie et archives sonores et présentation du projet Telemeta. Entretiens avec Pribislav Pitoëff et Joséphine Simonnot", *Transposition. Musique et sciences sociales*, 2011 (1).http://transposition-revue.org/article/ethnomusicologie-et-archives. Dernière consultation : 16/02/2012.

[7] Cannam, C., Landone, C. & Sandler, M. "Sonic Visualiser: An Open Source Application for Viewing, Analysing, and Annotating Music Audio Files", *Proceedings of the ACM Multimedia 2010 International Conference*. L'article et le logiciel sont disponibles en téléchargement gratuit : http://www.sonicvisualiser.org

[8] Clayton, M. R. L. "Observing entrainment in music performance: Video-based observational analysis of Indian musicians' tanpura playing and beat marking", *Musicae Scientiae* 11/1, 2007, 27–59.

[9] Clayton, M. R. L. "Time, Gesture and Attention in a Khyal Performance", *Asian Music* 38/2, 2007, 71-96.

[10] Cooper, D. & Sapiro, I. "Ethnomusicology in the Laboratory: From the Tonometer to the Digital Melograph", *Ethnomusicology Forum* 15/2, 301-313.

[11] Couprie, P., iAnalyse : un logiciel d'aide à l'analyse musicale », *Actes des Journées d'Informatique Musicale 2008 (JIM08)* : http://www.gmea.net/upload/17_PCouprie-JIM08.pdf

[12] Couprie, P. « Utilisation avancée du logiciel iAnalyse pour l'analyse musicale », *Actes des Journées d'Informatique Musicale 2010 (JIM10)* : http://jim10.afim-asso.org/actes/43couprie.pdf

[13] Fales, C. "The Paradox of Timbre", *Ethnomusicology* 46/1, 2002, 56-95.

[14] Fales, C. & McAdams, S. "The Fusion and Layering of Noise and Tone: Implications for Timbre in African Instruments" *Leonardo Music Journal* 4, 1994, 69-77.

[15] Gerischer, C. "*O Suingue Baiano*: Rhythmic Feeling and Microrhythmic Phenomena in Brazilian Percussion", *Ethnomusicology* 50/1, 2006, 99-119.

[16] Greene, P. & Porcello, T. *Wired for sound: engineering and technologies in sonic cultures*, Wesleyan University Press, Middletown, 2005.

[17] Greene; P. "Nepal's "Lok Pop" Music: Representations of the Folk, Tropes of Memory, and Studio Technologies" *Asian Music* 34/1, 2002-2003, 43-65.

[18] Guyot, P. *Réalisation d'une application informatique pour l'analyse des échelles musicale de chants traditionnels du Sud de l'Italie*, Rapport de stage dans le cadre du Master ATIAM (Acoustique, traitement du signal et informatique appliqués à la musique), Université Pierre et Marie Curie Paris VI, 2010. www.atiam.ircam.fr/Archives/Stages0910/Guyot.pdf

[19] Jairazbhoy, N. A. "The "Objective" and Subjective View in Music Transcription", *Ethnomusicology* 21/2, 1977, 263-273.

[20] Lartillot, O. & Toiviainen, P. "A Matlab Toolbox for Musical Feature Extraction From Audio", *Proceedings of the 10th International Conference on Digital Audio Effects* (Bordeaux), 2007. La Toolbox et le logiciel sont disponibles en téléchargement gratuit : https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox

[21] Lévy, F. "PARETO. Patchs d'Analyse et de Resynthèse des Echelles dans les musiques de Tradition Orale », Présentation en ligne. http://www.fabienlevy.net/Compositions/Pareto.html

[22] Lysloff, R. & Gay, L. (eds). *Music and technocultures*, Wesleyan University Press, Middletown, 2003.

[23] Mallet, J. "Ethnomusicologie des "jeunes musiques", *L'Homme*, 171-172, 2004, p. 477-488.

[24] Malm, K. "The Music Industry", in Helen Myers (ed.). *Ethnomusicology. An Introduction*. London: Macmillan, 1992, 349-364.

[25] Marandola, F. "The Study of Musical Scales in Central Africa: The Use of Interactive Experimental Methods", in Uffe Kock Wii (éd.), *International Symposium Computer Music Modeling and Retrieval (CMMR) Montpellier, France, May 26-27, 2003*), Berlin-Heidelberg: Springer, 2004, 34-41. (Coll. "Lecture Notes in Computer Science 2771").

[26] Martin, D.-C. "Les musiques en Afrique, révélateurs sociaux", *Ceras - Revue Projet n°283*, Nov. 2004. http://www.ceras-projet.com/index.php?id=1257. Dernière consultation : 16 février 2012.

[27] Morel, D. [Logiciel de description et d'aide à l'analyse des monodies *Monika*], *Patrimoines et Langages Musicaux,* Université Paris-Sorbonne. http://www.plm.paris-sorbonne.fr/spip.php?rubrique188

[28] Ndour, S. (ed.). *L'industrie musicale au Sénégal. Essai d'analyse*, Conseil pour le développement des sciences sociales en Afrique, Dakar, 2008.

[29] Schneider, A. "Sound, Pitch, and Scale: From "Tone Measurements" to Sonological Analysis in Ethnomusicology", *Ethnomusicology* 45/3 2001, 489-519.

[30] Stobart, H. & Cross, I. "The Andean Anacrusis? Rhythmic Structure and Perception in Easter Songs of Northern Potosí, Bolivia", *British Journal of Ethnomusicology* 9/2, 2000, 63-92.

[31] Suchoff, B. "Computerized folk song research and the problem of variants", *Computers and the Humanities*, 1967, 2/4, 155-158.

[32] Suchoff, B. "Some Problems in Computer-Oriented Bartokian Ethnomusicology", *Ethnomusicology* 13/ 3, 1969, 489-497.

[33] Tourny, O. "Le système scalaire des chants liturgiques éthiopiens", *Musicae Scientiae (Forum de Discussion 1 "Le syndrome du pentatonisme africain")*, 2000, p. 25-33.

[34] Tzanetakis, G., Kapur A., Schloss W. A. & M. Wright. "Computational Ethnomusicology" *Journal of Interdisciplinary Music Studies* 1(2), 2007, 1-24.

[35] Vassilakis, P. N., "Auditory roughness as a means of musical expression", *Selected Reports in Ethnomusicology* 12, 119-144

[36] Voisin, F. "Modélisation des systèmes d'accord des xylophones centrafricains", *Analyse Musicale* 23, 1991, 42-46.

[37] Voisin, F. "Musical Scales in Central Africa and Java: Modeling by Synthesis", *Leonardo Music Journal* 4, 1994, 85-90.

[38] Weisser, S. "Transcrire pour vérifier : le rythme dans les chants de *bagana* d'Ethiopie", *Musurgia* XIII/2, 2006, 51-62.

DU SAUVETAGE À LA PRÉSERVATION DES ŒUVRES MUSICALES CRÉÉES AVEC DISPOSITIF NUMÉRIQUE

Antoine Vincent
Univ. de Tech. de Compiègne
Heudiasyc UMR 7253 CNRS
& IRCAM
antoine.vincent@hds.utc.fr

Alain Bonardi
Université Paris 8
EA 1572 – CICM
& IRCAM
alain.bonardi@ircam.fr

Francis Rousseaux
Univ. de Reims Champagne-Ardenne
CReSTIC EA 3804
& IRCAM
francis.rousseaux@univ-reims.fr

RÉSUMÉ

La préservation des œuvres musicales était ces derniers siècles conditionnée par le recours à un substrat culturel stable, c'est-à-dire la partition, une organologie bien identifiée avec la classification des instruments et la lutherie qui permettait de fabriquer ces instruments. L'apparition des systèmes électroniques et numériques a chamboulé le monde de la composition, lui faisant perdre ses traditions pluri-séculaires et mettant en péril la préservation de la musique contemporaine. Devant la nécessité de mettre à jour les œuvres pour lutter contre l'obsolescence technologique, il nous paraît très intéressant d'étudier les processus de composition desquels nous pouvons extraire des informations pertinentes, nécessaires pour ressaisir certaines intentions originales du compositeur. Notre approche se fonde sur la création d'un modèle capable de représenter ce processus de production qui sera utilisé dans le projet ANR Gamelan au sein d'un ensemble d'outils permettant de le visualiser et de l'étudier. Nos premiers résultats suite à une phase d'immersion dans les pratiques actuelles de composition concernent la représentation de certaines informations extraites d'une production et qu'il nous semble déjà vital de posséder lors d'une migration d'une œuvre afin d'en préserver l'authenticité.

1. INTRODUCTION

Les pratiques de composition s'adaptent aux technologies du moment, et parfois les façonnent. Elles évoluent en phase avec leurs instruments. Au cours du temps, des œuvres naissent, vivent et meurent. Certaines perdurent, notamment grâce à leur valeur symbolique ou leur popularité, qui entraînent une transmission du savoir-faire permettant leur reproductibilité. D'une tradition fortement ancrée dans la culture, la musique était jusqu'au siècle dernier transmise en utilisant la partition, substrat culturel stable permettant de s'abstraire du son pour retourner vers lui grâce à l'instrument.

Désormais, les nouvelles technologies ont transformé l'approche de la représentation musicale classique : les « instruments » matériels ou logiciels deviennent obsolètes très rapidement et il n'est pas toujours possible de s'en abstraire pour rejouer une pièce telle qu'imaginée par son compositeur. Il naît ainsi un réel problème de préservation qui met en danger la musique, certaines œuvres étant proches de la disparition, voire, déjà disparues.

Notre contribution portera sur l'approche que nous souhaitons développer pour créer une nouvelle forme de représentation du processus compositionnel à base de nouvelles technologies. Nous commencerons par présenter le contexte de la production musicale actuelle, puis dans une deuxième partie, nous aborderons les objectifs de préservation recherchés, avant de continuer sur notre étude terrain et notre immersion dans ce monde de la création sonore. Enfin, la quatrième et dernière partie, avant de conclure, présentera nos premiers résultats dans l'élaboration d'un langage de représentation des processus de composition sonore.

2. CONTEXTE DE LA PRODUCTION MUSICALE

2.1. Évolution des pratiques de composition

Avec l'apparition des systèmes numériques, le compositeur a fait évoluer ses pratiques de création des œuvres musicales. Il ne se contente plus de travailler uniquement les notes mais rapproche davantage sa pratique de composition des sons eux-mêmes en les manipulant directement : « nous vivons, en ce début de XX^{ème} siècle, un véritable bouillonnement organologique, où instruments acoustiques, technologies analogiques et technologies numériques issues de l'informatique forment un système de plus en plus intégré » [7].

Cette idée de lien fort entre évolution des technologies et des pratiques de composition est illustrée par exemple par la relation qu'ont entretenu Miller Puckette et Philippe Manoury et la création de *Jupiter* en 1987 : l'œuvre a nécessité le développement d'un programme spécifique, et c'est suite à la création de l'œuvre de Philippe Manoury que Miller Puckette a sorti la première version du logiciel Max ; ainsi chacun est né de l'autre [5].

Ces nouvelles pratiques musicales posent le problème de la préservation des œuvres, car elles s'éloignent fortement des piliers séculaires du corpus théorique classique :

– l'écriture musicale abstraite telle qu'elle est pratiquée dans la musique occidentale, avec sa double organisation des hauteurs et des durées ;

– l'organologie, qui propose une classification des instruments selon deux dimensions : par voix (au sens des quatre voix héritées de la musique vocale) et par mode d'interface au son (par exemple les cordes frottées, pincées ; les anches simples, doubles ; les cuivres, etc.).

A contrario de la musique contemporaine avec dispositif électronique dont la durée de vie est fortement liée à l'obsolescence technologique de plus en plus rapide et n'ayant plus de représentation abstraite stable de la partie électronique, une œuvre comme la *Sonate en la mineur, D. 821* de Franz Schubert, écrite en 1824 pour arpeggione et piano, peut encore être jouée aujourd'hui. L'apeggione a presque disparu : grâce à la partition, une transposition a été réalisée, l'œuvre continue sa vie et est transmise, désormais jouée au violoncelle ou à l'alto.

La communauté informatique a régulièrement tendance, face à cette perte de substrat culturel stable, à se réfugier derrière le code source des programmes utilisés. Or, nous n'avons pas de visibilité sur la pérennité d'un langage en informatique, même si celui-ci se trouve être normalisé. De plus, même si nous conservons ce code, nous n'aurons pas forcément le même rendu sonore : nous savons déjà qu'un programme rédigé en C [1] peut produire des résultats sonores différents en fonction du processeur et du compilateur utilisés [2].

2.2. Difficile transmission des œuvres et des pratiques...

En l'absence de ces substrats dits stables, la transmission de ces nouvelles formes d'expression et des savoirs qui y sont liés pose problème. Premièrement pour la rejouabilité même de l'œuvre, d'autre part pour la compréhension des pratiques de composition. Ainsi l'enseignement académique autour de ces notions n'en n'est encore qu'à ses prémisses : les formations techniques autour de ce que nous appelons l'informatique musicale se développent, mais l'enseignement musical en lui-même, autour de l'écriture et de l'organologie des dispositifs électroniques, est encore balbutiant. Car comment transmettre un savoir si nous n'avons aucune forme stabilisée de l'œuvre et du contexte de production, sans écriture et en utilisant des outils très éphémères à l'échelle des générations humaines ?

Nous n'avons actuellement aucune réponse générale, mais parfois ponctuelle. Par exemple, dans le domaine des musiques savantes : *Diadèmes* de Marc-André Dalbavie, créée en 1986 à l'aide de synthétiseurs TX816 devenus aujourd'hui obsolètes. Pour une reprise organisée en 2008 aux États-Unis, un sampleur logiciel basé sur des échantillons exportés du synthétiseur original a été utilisé [4]. Or, cette mise à jour de l'œuvre via cette technique particulière a été rendue possible grâce à deux critères :

– la transmission orale directe : les réalisateurs en informatique musicale, travaillant directement sur le montage des œuvres et en collaboration avec les compositeurs détiennent énormément de connaissances ;
– la transmission écrite indirecte : l'annotation de partitions ou toute autre inscription d'un acteur impliqué dans la phase de composition ou de création de l'œuvre peut devenir une source très importante d'information lors d'un processus de mise à jour.

Pour *Diadèmes*, une phase de validation avec le compositeur a aussi été possible, ce qui n'est malheureusement pas toujours le cas.

2.3. ...et donc une préservation complexe

Nous avons déjà cité le réalisateur en informatique musicale (surnommé le RIM) : à l'IRCAM, il est le médiateur entre le compositeur et le système numérique. Il a notamment en charge le développement d'outils pour une production en particulier et le pilotage des logiciels durant la performance lors des concerts. Mais le RIM devient dorénavant un producteur d'archives : il songe dès le début à la préservation des œuvres et à sa possible rejouabilité afin de la faire vivre dans le temps, notamment en l'adaptant aux nouvelles conditions de restitution.

Le RIM manque d'outils pour stabiliser l'objet musical dans un format universel qui permettrait une nouvelle exploitation sans faire de modification profonde. Ainsi pour remonter l'œuvre, il va chercher à récupérer un maximum d'informations (logicielles, matérielles, temporelles, spatiales, etc.) pour effectuer les transformations, en prenant en compte le respect de l'identité de l'œuvre en tentant de suivre les intentions auctoriales qu'il arrivera à saisir. Dans le cas de *Diadèmes* présentée précédemment, Serge Lemouton, RIM à l'IRCAM, a eu en charge le portage de l'œuvre seize ans après sa dernière exécution.

Une question émerge : comment conserver l'authenticité d'une œuvre à chaque mise à jour ? Cette question avait trouvé réponse dans le cadre de la musique baroque et le cas de résurrections d'œuvres restées plusieurs siècles sans être interprétées, mais se pose de nouveau dans le domaine de la musique faisant appel aux technologies numériques. Le musicologue souhaite mettre en perpective des intentions de l'auteur, qui doivent être respectées au plus près durant la vie complète de l'œuvre, et pour cela il faut offrir lors des migrations, les éléments pertinents pour réaliser le portage dans les meilleurs conditions possibles.

3. OBJECTIF DE PRÉSERVATION

3.1. Viser la rejouabilité de l'œuvre

Nous avons présenté en ce début d'article les nécessaires mises à jour technologiques des sources de production qu'il faut opérer pour faire face à l'obsolescence des équipements et qui peuvent mettre les œuvres en danger : en effet, il ne suffit pas de simplement trouver des équivalents au niveau du résultat sonore, il faut avant tout que les transformations aient un sens musical et culturel. Mais il est évidemment nécessaire de faire ce travail de remobilisation des œuvres pour lutter contre l'absence de stabilité

1 . Le langage C est un langage de programmation impératif très utilisé, créé dans les années 1970 et servant de base au système UNIX et à d'autres langages de programmation.

des instruments numériques qui touchent tous les répertoires les utilisant, que ce soit sous la forme de boîtier matériel ou de logiciel.

Les intentions sont maintenant directement associées aux traces numériques de l'activité de composition mais restent impossibles à coder informatiquement. Il n'est possible de les resaisir que par reconstruction à chaque fois que nous le souhaitons, comme pour remonter une œuvre. Afin d'aider cette construction, l'idée est d'être capable de fixer l'activité de création sonore dans laquelle se trouvent implicitement certaines de ces intentions, en définissant le bon niveau d'abstraction à appliquer : assez général pour se dégager de la contingence des outils mais pas trop abstrait pour avoir des informations utilisables [1].

Notre idée est de travailler la préservation en offrant une représentation du processus de production de l'œuvre musicale et des informations liées, à partir des traces pertinentes que nous pouvons extraire durant l'acte de composition. La première difficulté, dès que nous évoquons cette représentation, réside dans le fait que l'œuvre est considérée comme un objet unique. Il faudra ainsi élaborer un modèle qui couvrira différents types d'objet, pour qu'il permette de préserver les informations pertinentes afin d'en offrir une représentation utilisable lors de chaque modification souhaitée sur l'œuvre.

3.2. Traçabilité des informations

Notre approche consiste à ne pas chercher d'alternative aux migrations perpétuelles et régulières, qui restent pour nous la référence en matière de préservation, car c'est une recherche qui ne donne actuellement aucun résultat viable. Tel est l'objectif du projet ANR [2] Gamelan [3] : concevoir et réaliser une maquette d'un méta-environnement permettant de visualiser selon plusieurs échelles le cycle de vie d'une œuvre et des processus de production déployés durant sa réalisation.

Cet environnement permettra principalement la représentation de l'œuvre et des informations associées nécessaires pour la rejouer. Nous aurons ainsi un archivage intelligent des objets, c'est-à-dire permettant de remonter à la source de la production, afin de comprendre les processus qui auront permis sa création. Cette approche est principalement fondée sur la provenance des informations et la qualité des traces qu'il est possible d'extraire. Partenaires du projet, EMI Music France, le Groupe de Recherches Musicales de l'INA et l'IRCAM, ont tous trois des traditions de composition différentes (cf. figure 2) : depuis la création d'un groupe en vue d'une diffusion sur CD, jusqu'à la musique mixte en passant par les œuvres concrètes ; ils fournissent des cas d'usages permettant l'élaboration de l'environnement.

2 . Le projet Gamelan est financé par l'agence nationale de la recherche et a début en 2009 pour se terminer en 2013. Partenaire du projet : Heudiasyc UMR 7253 CNRS UTC, IRCAM, INA-GRM, EMI Music France.

3 . Gamelan : un environnement pour la Gestion et l'Archivage de la Musique et de L'Audio Numériques.

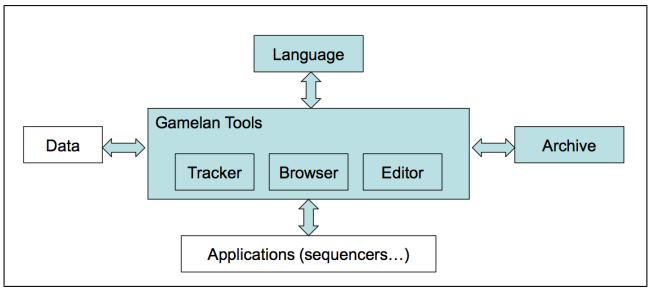


Figure 1. Architecture globale de l'environnement Gamelan.

La figure 1 présente l'architecture globale de cet environnement, fondée autour des Gamelan Tools et au nombre de trois :

- le tracker, exécuté en tâche de fond sur l'outil utilisé par le compositeur, capable de suivre les opérations effectuées afin de capter les étapes du processus de production ;
- le browser, outil utile pour visualiser le flux d'une production ;
- l'editor, qui permettra de compléter un processus capté.

Les cibles principales de cet environnement sont les réalisateurs en informatique musicale ou les ingénieurs du son qui voudraient remonter une œuvre. Ils pourront ainsi visualiser la production, identifier les acteurs présents et localiser les documents qu'ils souhaiteraient consulter ; pour au final déterminer quels outils seront les plus adaptés pour effectuer la mise à jour. Les musicologues pourront aussi être intéressés, notamment pour étudier les pratiques de composition dans un but pédagogique, en démontant et remontant les œuvres par exemple.

Tous ces outils sont nécessaires mais non suffisants car il ne faut pas uniquement préserver les informations qu'il est possible de capter lors de la production, il est aussi important d'assurer un accès cohérent et logique aux informations. Dès lors, il est nécessaire d'avoir un langage au sein de l'architecture de la figure 1 qui permette de modéliser les processus de production.

3.3. Modélisation des processus de production

L'objectif principal du projet Gamelan est l'élaboration d'un modèle offrant un moyen de représenter les informations d'une production musicale. Lorsque nous évoquons la notion de langage de modélisation, nous sous-entendons un langage formel de représentation et non un langage purement informatique, et nécessitant donc la mise en œuvre d'un vocabulaire spécifique qui sera élaboré à partir d'une ontologie du domaine.

Dans le milieu artistique, nous l'avons déjà abordé, l'objet final est un prototype, non dans le sens de la pensée courante où le prototype est le premier exemplaire en vue d'une production de masse, mais dans sa définition littérale de premier objet, d'« exemple le plus parfait, le plus exact ». Ainsi les artistes envisagent les œuvres comme le

Type de musique	Définition	Étude	Institution représentative
Acoustique Enregistrement	Musique traditionnelle au sens classique (sur partition) et issue d'enregistrement en studio (pour la création d'un album par exemple).	Entretiens Directeurs artistiques Ingénieurs du son Suivi de production Œuvre classique	EMI Music France
Acousmatique	La production musicale effectuée au travers de manipulation de sons existants offre en sortie une œuvre sur un support fixe.	Entretiens Compositeurs Suivi de production Mise à jour d'œuvre	INA-GRM
Mixte	Elle intègre dans une même représentation des éléments issus de la synthèse sonore ou des transformations temps-réels et les performances effectuées par des musiciens sur des instruments traditionnels.	Entretiens Réalisateurs en informatique musicale Suivi de production Migration œuvre	IRCAM

Figure 2. Les différents terrains d'étude.

résultat final « idéal et expérimental » [3] et non comme une étape de création en série. Cette unicité est un problème dans le travail de modélisation car chaque création et surtout chaque processus étant différents, nous devons trouver le juste niveau permettant de couvrir plusieurs méthodes de production sonore.

Cette problématique de représentation des œuvres musicales est une question classique, qui se complexifie avec l'apparition de nouveaux modes de production qui n'offrent pas de représentation classique comme la partition. L'objectif ici est d'offrir cette description en travaillant sur le niveau d'abstraction utilisé : nous cherchons à nous abstraire des codes informatiques pour lutter contre l'obsolescence technologique sans toutefois rester dans une description en langue naturelle, bien trop imprécise pour livrer les informations souhaitées. Nous cherchons donc à repérer les invariants du contenu et à ne pas conserver les moyens permettant la reproduction. Cette problématique du « niveau des connaissances » [6] est typique du domaine de l'ingénierie des connaissances, mais notre approche puise son originalité dès l'étape d'acquisition de ces connaissances : nous ne disposons pas de corpus de document à exploiter et nous devons commencer par l'élaborer nous-même, via une étape d'immersion au cœur de la production musicale actuelle [8].

4. SUIVI DE PRODUCTION : AU CŒUR DES PROCESSUS

4.1. Sélection des œuvres

Afin d'élaborer notre propre corpus, qui servira de base pour la modélisation et la validation du résultat, nous ciblerons différentes productions ou mises à jours effectuées ou en cours afin de maximiser la couverture de notre étude. Nous pouvons retrouver dans la figure 2 un relevé des types d'œuvres effectué au début du projet nous permettant d'aiguiller nos choix.

Ainsi nous nous intéresserons notamment à *Saturne* d'Hugues Dufourt, dont une migration vient d'être effectuée par Yann Geslin au sein du GRM. Cette œuvre a été créée en 1979 au Centre Pompidou et fut écrite pour vingt-deux musiciens, mélangeant instruments classiques à vent et les percussions aux guitares électriques et orgues électriques de tradition populaire. L'œuvre a été mise à jour plusieurs fois, avec certaines transformations majeures : par exemple, en 1991, elle a été portée sur un synthétiseur programmable. C'est une œuvre intéressante pour nous car en 2006, elle avait quasiment disparu : un concert a dû être annulé car le re-montage de l'œuvre avait échoué. Pour la sauver, Yann Geslin, en 2010, dû repartir d'une version ancienne, celle de 1991 et la porter sur des environnements numériques. Et pour réussir cette tâche, il a réalisé un travail que nous pourrions qualifier d'archéologique en cherchant toutes les informations possibles auprès des différentes personnes qui sont travaillé sur *Saturne* durant son cycle de vie. La mise à jour étant effectuée, cette nouvelle contribution permet de prolonger la durée de vie de l'œuvre.

4.2. Zoom sur *Liszt voyageur*

À côté d'une œuvre musicale savante, utilisant les nouvelles technologies, nous nous intéressons aussi aux enregistrements. Ce type de production, même s'il est fixé sur support, présente deux aspects intéressants :

- les morceaux d'un CD font régulièrement l'objet de repurposing, comme une reprise pour être utilisés sur un DVD, ou d'arrangement pour être repris dans le cadre de publicités ; elles ont ainsi elles aussi un cycle de vie et des processus de transformation ;
- elles possèdent des phases de production (enregistrement, montage, mastering) bien distinctes, qui peuvent être analysées car présentant des parties

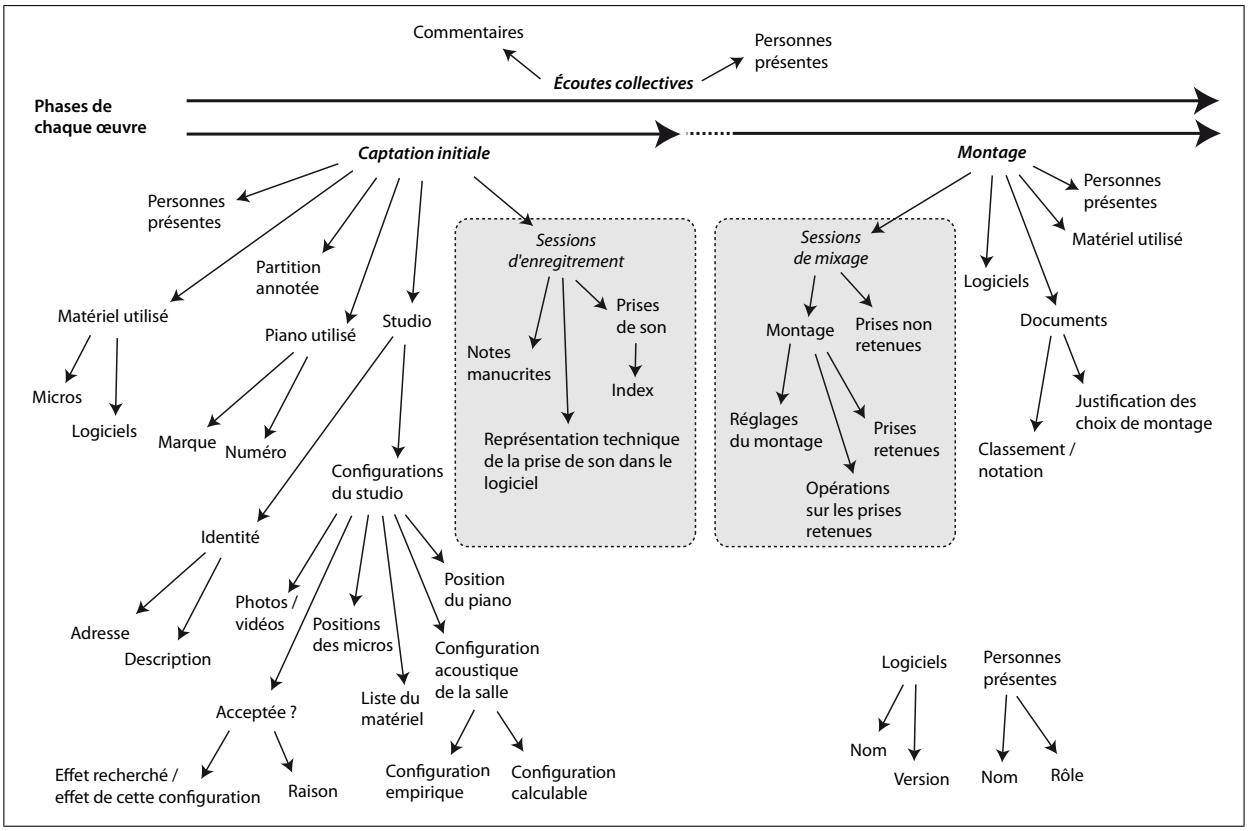


Figure 3. Informations extractibles des trois phases de la production du CD *Liszt voyageur*.

communes faisant intervenir différents acteurs, et souffrant ainsi de problèmes de représentation et de transmission.

Nous nous sommes donc intéressés à la production du CD de piano classique *Liszt voyageur*, enregistré par la pianiste Emmanuelle Swiercz à l'occasion du bicentenaire de la naissance de Franz Liszt. Notre choix trouve aussi son origine dans le fait que cette production utilise la plateforme ProTools, standard largement utilisé dans la production contemporaine, qui ne possède pas de représentation des étapes d'élaboration. Ce logiciel propriétaire est très verrouillé et ne permet pas d'extraire d'information autour du processus de production.

Nous avons ainsi eu l'occasion de nous intégrer dans les différentes phases de la production de cet enregistrement afin de collecter un maximum d'informations et lancer une réflexion sur celles qui paraissent pertinentes de préserver. La figure 3 représente ces informations qui donnent des indications sur la production de l'œuvre, notamment sur les choix effectués ce qui permet d'appréhender certaines intentions de la pianiste, mais aussi du directeur artistique [9].

Nous nous intéressons à deux phases de la production : la captation initiale, suivie de la phase de montage. Les écoutes collectives sont une activité transversale qui a lieu durant toute la phase de production. Pour chacune des phases, nous nous sommes efforcés d'extraire les informations qui ont influencé la création du CD et permettent la compréhension des choix effectués. Par exemple, il nous paraît important de connaître les personnes présentes, le piano utilisé, la configuration du studio (l'enregistrement a eu lieu à l'espace de projection de l'IRCAM, hautement configurable), ainsi que tous les documents comme les partitions annotées ou des photographies réalisées durant la production. Les données numériques, comme les sessions d'enregistrement et de mixage, sont très importantes car elles contiennent toutes les étapes techniques réalisées.

Cette liste d'informations extractibles du processus est loin d'être exhaustive. Mais elle permet de répondre à certains critères de ce que nous attendons dans la préservation de l'œuvre finale. Ainsi, elles permettent de resaisir certaines intentions de la pianiste et de la direction artistique durant le projet : durant la phase de captation, nous avons le rendu sonore recherché via la configuration de la salle et le matériel utilisé, et durant la phase de montage, nous trouvons la documentation concernant les choix opérés au travers du mix final.

Cette première représentation nous permet déjà de proposer un premier tri des informations et une manière de les visualiser. Il reste énormément d'informations qui ne relèvent pas uniquement du contexte, notamment dans les fichiers de session, qui permettent de connaître l'ensemble des étapes menant au résultat final, ou qui se trouvent en dehors du système numérique que les intervenants pourraient ajouter pour compléter la description.

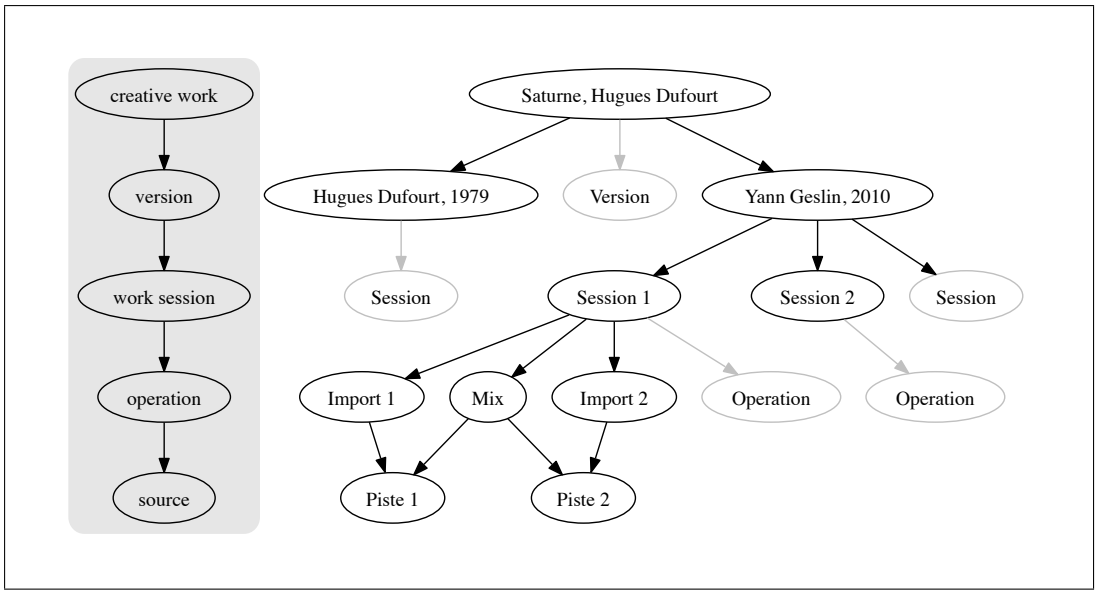


Figure 4. Graphe de décomposition d'une œuvre.

5. MODÉLISATION DES PROCESSUS DE PRODUCTION SONORE

5.1. Qu'est-ce que la modélisation ?

Détaillons cette notion de modélisation. Nous souhaitons proposer une représentation du processus de production, avec un juste niveau permettant de nous abstraire du code informatique (et éviter la trop forte dépendance à une technologie de plus en plus rapidement obsolète) mais assez prescriptif pour présenter les différentes étapes du cycle de vie de l'œuvre et les étapes d'élaboration et de transformation de l'œuvre. Afin d'exemplifier, tentons de décomposer très simplement une œuvre musicale sur la figure 4.

Sur ce graphe, nous pouvons visualiser à gauche une décomposition d'une œuvre et à droite nous reprenons l'exemple de *Saturne* de Hugues Dufourt. Ainsi, le pattern retenu est qu'une œuvre est constituée d'un ensemble de versions (dans notre exemple nous avons la version originale d'Hugues Dufourt et un ensemble d'autres versions, dont la dernière de Yann Geslin qui résulte de différentes mises à jour).

Pour chaque version, nous retrouvons un ensemble de séances de travail afin d'élaborer une version de l'œuvre (version d'origine ou version mise à jour). Ces séances de travail ne sont pas nécessairement liées à une session d'un logiciel de montage mais représente une unité temporelle, choisie par le compositeur (par exemple, une séance de travail pourrait être une journée ou une demi-journée de travail). Dans chaque séance de travail, nous réalisons un certain nombre d'opérations, et chaque opération possédera potentiellement une ou plusieurs sources. Par exemple, dans une séance de travail, nous pourrions avoir deux imports de son dans le logiciel de montage puis une action de mixage.

Nous comprenons, avec cette première explication issue d'un simple graphe de décomposition, que nous souhaitons être capable de connaître l'enchaînement des actions qui ont été effectué dans chaque version et ainsi dresser une génétique de l'œuvre, pour espérer retrouver le processus de production.

5.2. Proposition simplifiée d'une modélisation

Nous proposons en figure 5 un extrait de cette taxinomie des concepts plutôt que de la présenter en détail, intentionnellement lacunaire et perdant son engagement sémantique pour détailler le résultat souhaité quand nous parlons d'une ontologie référentielle.

Nous retrouvons ici sous la forme de trait en plein l'arbre des concepts, les traits grisés présentant des individus que nous avons ajouté pour peupler le modèle et en trait disjoints les relations que nous pouvons poser entre plusieurs concepts.

Cet extrait présente des bribes de la production de *Nuages gris* (œuvre enregistrée pour le CD *Liszt voyageur*). Ici par exemple nous pouvons reconstituer :

- *Nuages Gris* est l'œuvre ;
- *Session 1 Nuages Gris* est un fichier de session ;
- *Session 1 Nuages Gris* est un élément de l'œuvre ;
- *Emmanuelle Swiercz* participe à *Session 1 Nuages Gris* ;
- *Emmanuelle Swiercz* joue du *Piano* ;
- *Session 1 Nuages Gris* a été enregistré à l'*Espace de Projection* ;
- la *Piste 1* appartient à *Session 1 Nuages Gris* ;
- la *Région 1* appartient à la *Piste 1* ;
- etc.

Pour des raisons de lisibilité, très peu de choses ont été représentées, mais nous voyons tout de suite la puissance et la complexité d'un tel outil de représentation, qui une fois peuplé, peut stocker énormément d'informations.

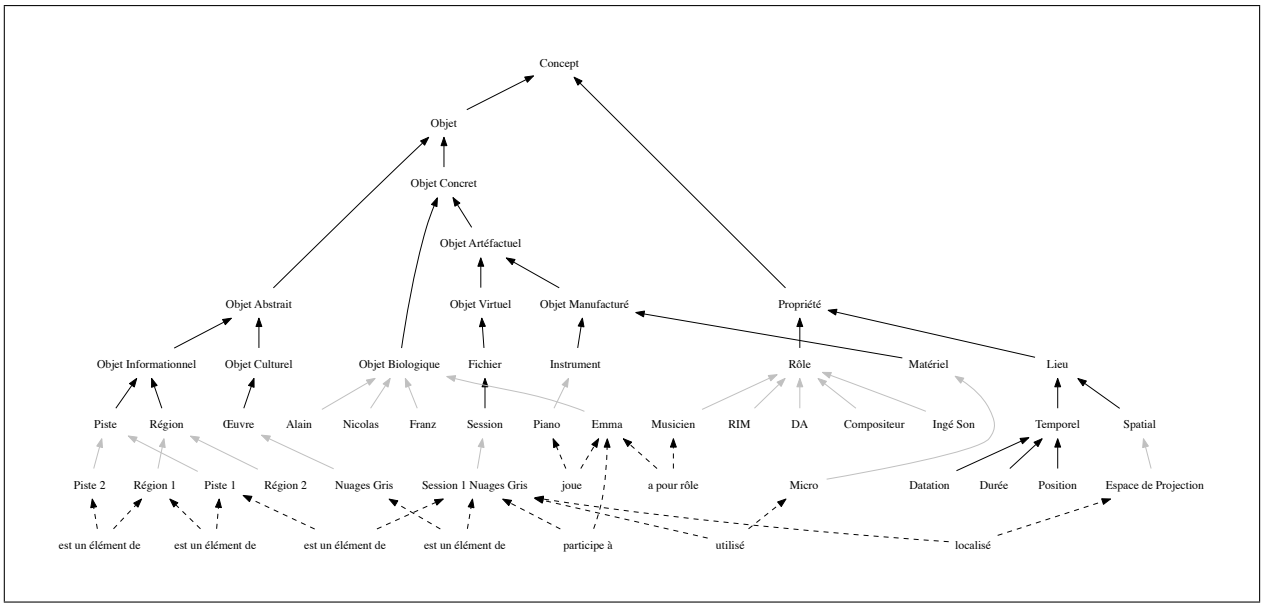


Figure 5. Extrait instancié avec *Nuages gris* d'une possible ontologie des processus de production sonore.

5.3. Utilisation et utilité du modèle de représentation

Nous montrons ici sur un exemple comment les modélisations explicitées dans les paragraphes précédents peuvent contribuer au processus de préservation et de ressaisissement de l'intention. L'extrait du modèle présenté à la figure 5 montrait une partie des concepts, une instanciation et certaines relations correspondant à la situation d'enregistrement de la pièce *Nuages gris* de Liszt dans le cadre du CD *Liszt voyageur* de la pianiste Emmanuelle Swiercz.

Imaginons un scénario d'utilisation, qui consiste à s'intéresser à la représentation et à la transmission d'information entre les trois phases de la production d'un CD (captation, montage et mastering) qui sont souvent bien séparées dans le temps. Ici, l'apport du projet Gamelan et du modèle permet de faire face à cet éclatement temporel entre les trois phases de production, et aux éventuels changements d'intervenants. Dans le cas de la pianiste Emmanuelle Swiercz, l'enregistrement s'est déroulé en trois jours fin décembre 2010, le montage en février et mars 2011, et le mastering en mai 2011. Ces périodes importantes d'interruption conduisent à de nécessaires oublis et demandent ensuite une remobilisation pour interpréter les objets numériques produits dans les phases précédentes.

La modélisation effectuée sur *Nuages gris* permet par exemple, au niveau du montage, de retracer l'historique de l'enregistrement pour remettre en perspective les différentes prises de son : dans cette production, chaque session ProTools était dédiée à une pièce du CD final, et non à une session de travail ; selon les souhaits de la pianiste et de la direction artistique, il était possible de revenir sur une session ProTools à d'autres moments de la période d'enregistrement. Plus largement, cette représentation des connaissances permettra de visualiser de manière synoptique l'ensemble de la production d'un CD sous plusieurs formes, que ce soit temporellement, par intervenant, par session ou sous la forme de flux de production avec les dépendances d'exportation/importation de fichiers entre sessions.

6. CONCLUSION

Ce premier travail d'immersion au sein des processus de production nous a permis d'étudier des pratiques de compositions actuelles. Le bilan de ce suivi de production a donné lieu à la mise en place de différentes modélisations que nous testons afin de les valider. La prochaine étape sera de mettre en place un langage de représentation des connaissances qui sera utilisable au sein de l'environnement développé dans le cadre du projet Gamelan, afin de classer automatiquement les données via le modèle.

Nous espérons que notre travail pourra à terme aussi devenir ou simplement influencer l'élaboration d'un modèle de stockage, ne visant non pas la pérennité de l'œuvre, mais une conservation efficace de tous les artefacts liées à l'œuvre et à sa production, assurant ainsi un accès à un maximum de documentation s'y rattachant.

7. REFERENCES

[1] Bachimont, B. *Archivage audiovisuel et numérique : les enjeux de la longue durée*, Chapitre 8, p. 195-222. Hermès, Paris, 2009.

[2] Collins, N. *Introduction to Computer Music*. Wiley, 2010.

[3] During, E. "Prototypes", Entretien avec Franck Madlener, *L'Etincelle*, Paris : Ircam, 2010.

[4] Lemouton, S., Ciavarella, R. Bonardi, A "Peut-on envisager une organologie des traitements sonores temps réel, instruments virtuels de l'informatique musicale ?", *Cinquième Conférence de Musicologie*

Interdisciplinaire (CIM'09), actes de la conférence, pages 118-121, Paris, octobre 2009.

[5] Manoury, P. *Considérations [toujours actuelles] sur l'état de la musique en temps réel*. Revue l'Etincelle, Prospectives, Paris : Ircam – Centre Georges Pompidou, 2007.

[6] Newell, A. "The Knowledge Level", *Artificial Intelligence*, 18 :87– 127.

[7] Stiegler, B. "Bouillonnements organologiques et enseignement musical", *Les dossiers de l'ingénierie éducative*, Paris, France, 2003.

[8] Vincent, A. *Préservation d'œuvres musicale : étude du processus de production*. Mémoire de Master de l'Université de Technologie de Compiègne, Compiègne, 2010.

[9] Vincent A., Bonardi, A., Bachimont, B. "Préservation de la musique avec dispositif électronique : l'intérêt des processus de production sonore", *Actes des 17èmes Journées d'Informatique Musicale (JIM 2011)*, Université Jean-Monnet Saint-Etienne, 25-27 mai 2011, p. 71-76.

UN SYNTHÉTISEUR DE LA VOIX CHANTÉE BASÉ SUR MBROLA POUR LE MANDARIN

Liu Ning
CICM (Centre de recherche en Informatique et Création Musicale)
Université Paris VIII, MSH Paris Nord
liuningchine@yahoo.fr

RESUME

Dans cet article, nous présentons le projet de développement d'un synthétiseur de la voix chantée basé sur MBROLA en mandarin pour la langue chinoise. Notre objectif vise à développer un synthétiseur qui puisse fonctionner en temps réel, qui soit capable de se synchroniser avec un séquenceur MIDI ou puisse être piloté par un clavier MIDI pour produire la voix chantée en mandarin. Le développement de l'application est basé sur un algorithme existant - MBROLA, ainsi l'objet « mbrola~ » est utilisé dans notre programmation. Notre travail consiste en la création de la première base de données en mandarin pour MBROLA, et le développement d'une application informatique musicale dans l'environnement Max 5.

1. INTRODUCTION

Aujourd'hui, les technologies de la synthèse de la voix parlée sont très diversifiées et avancées, et en même temps les synthétiseurs pour la voix chantée sont en cours de développement. Les synthèses sont basées sur des méthodes différentes, par exemple la synthèse FM de John Chowning [1]. Le MUSSE (Music and Singing Synthesis Equipment) de Johan Sundberg est un synthétiseur qui a intégré la synthèse par règle et la synthèse vocale formantique [2]. Le programme CHANT de Xavier Rodet est basé sur la synthèse formantique [3]. Le SPASM (Singing Physical Articulatory Synthesis Model) est développé par Perry Raymond Cook basé sur le modèle physique [4]. Le logiciel Vocaloid est développé par l'université Pompeu Fabra, et commercialisé par YAMAHA, il est basé sur la synthèse sinusoïdale [5].

En 2002, le synthétiseur de la voix chantée « On-The-Fly » a été proposé par le département de l'informatique de Tsing Hua Université à TAIWAN [6]. L'algorithme de PSOLA a été employé pour mettre en œuvre cette synthèse.

En 2004, le professeur GU Hong Yan et son collègue CHEN An Rui de l'Université Nationale de Science et de Technologie de Taiwan ont développé et amélioré une synthèse additive basée sur la méthode sinusoïdale. Ce système est capable de produire la voix chantée en fonction de la mélodie et des paroles à partir d'un fichier MIDI [7].

Néanmoins, les synthèses de la voix chantée pour la langue chinoise ont encore des limites techniques. Par exemple, le système à partir de la lecture de fichier MIDI est un système en temps différé. La concaténation de l'unité enregistrée basée sur le phonème donne une dis-naturalité à la voix. Les synthèses ne produisent pas le changement de la fréquence fondamentale linéaire (utile pour le *glissando* ou le *portamento*).

Conscients de ces limites constatées dans les synthétiseurs de la voix chantée pour la langue chinoise existants, et dans la perspective d'apporter des améliorations, nous allons parler dans cet article de notre développement d'une application basée sur le synthétiseur MBROLA via « mbrola~ ». Cette application sera réalisée pour une synthèse en temps réel. Les fonctions de la production de la voix *portamento* et de la voix *glissando* seront intégrées dans cette application. Dans un premier temps, nous présenterons la création d'une base de données diphonique en mandarin pour MBROLA ; ensuite nous présenterons la programmation de l'application dans l'environnement Max 5.

2. L'ALGORITHME MBROLA ET L'OBJET « mbrola~ » POUR MAX 5

MBROLA (Multi-Band Re-synthesis Overlap Add) [8] est un algorithme connu de synthèse vocale basé sur la synthèse concaténative de l'unité diphonique. Le synthétiseur MBROLA a l'avantage de pouvoir produire une voix très intelligible. Grâce aux travaux des équipes de chercheurs de différents pays, aujourd'hui, le système MBROLA est capable de produire la parole pour 35 langues différentes.

En 2005, Nicolas D'Alessandro, Raphäel Sebbe, Baris Bozkurt et Thierry Dutoit du Laboratoire TCTS de la Faculté Polytechnique de Mons (Belgique) ont développé l'objet « MaxMBROLA~ » basé sur le synthétiseur MBROLA pour l'environnement programmation Max/MSP [9]. L'objet « MaxMBROLA~ » et «mbrola~ » (la nouvelle version de l'objet « mbrola~ »

est sortie en 2010) permet de rendre le fonctionnement du synthétiseur MBROLA en temps réel. « mbrola~ » a été développé pour Max 5. Nous pouvons utiliser cet objet pour charger une base de données MBROLA, modifier la durée originale de la séquence de la parole, varier la fréquence fondamentale de la voix, etc.

3. L'ENVIRONNEMENT MAX 5

Conçu par l'IRCAM et Cycling'74, Max 5 est destiné aux artistes, aux musiciens, aux designers sonores, aux enseignants ou encore aux chercheurs. Ce langage de programmation visuelle, proche de la programmation orientée objet donne l'accès à des nombreux modules et fonctions prédéfinies, et à des instructions pouvant être stockées pour faciliter l'assemblage au sein d'une interface graphique. Max 5 permet à l'utilisateur de faire des manipulations interactives, grâce au calcul et au traitement du signal audio en temps réel [10].

4. CRÉATION DE LA PREMIÈRE BASE DE DONNÉES MBROLA POUR LE MANDARIN

Pour la création de la base de données MBROLA, Nous devons d'abord trouver toutes les combinaisons diphoniques possibles et nécessaires pour la voix chantée en mandarin. Notre édition de la liste des diphones est principalement basée sur les règles de transcription phonétique PIN YIN [11]. 410 segments diphoniques étaient sélectionnés pour la base de données de la voix parlée. Nous savons que dans le chant, nous pouvons répéter la dernière voyelle pour prolonger une syllabe chinoise. Cependant, certaines voyelles ou combinaisons de voyelles ne sont pas présentées dans le dictionnaire chinois. Par exemple, dans notre figure1 ci-dessous, la dernière syllabe est prolongée par une prononciation /ei/, alors que cette prononciation propre à la voix chantée n'existe pas dans le dictionnaire pour la voix parlée. C'est la raison pour laquelle, après avoir effectué des analyses phonétiques et modifié certaines règles de prononciation de la voix parlée pour les adapter à la voix chantée, nous avons complété la base de données de la voix parlée en y ajoutant un certain nombre de prononciations qui n'existent que dans la voix chantée. Grâce à ces démarches, la liste des diphones de la voix parlée comprend finalement 519 segments au lieu de 410.

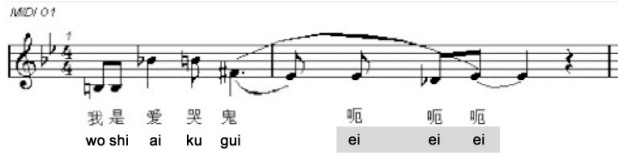


Figure 1. Extrait de chanson « Ai Ku Gui »

Le prélèvement des échantillons de la voix a été fait avec des matériels audio professionnels, la qualité originale de prélèvement de la voix est en 16bit 44.1kHz.

Pour avoir des prononciations parfaites de toutes les syllabes chinoises, deux chanteuses dont le caractéristique de la voix est proche ont participé dans les enregistrements. Lors de l'enregistrement, nous avons veillé à ce que la hauteur des prononciations soit constante. Après la MBROLISATION, le timbre de la voix est très acceptable.

Dans la phase de segmentation, nous avons bouclé les segments de voyelles afin de garantir la continuité dans le chant. Après beaucoup d'essais, nous avons également déterminé les durées générales des segments : la durée de l'initiale est environ 60ms et la durée de la finale est 200ms. Si dans le synthétiseur ces règles de durée sont respectées, la production de la syllabe chinoise peut être clairement audible, et le temps de réaction du changement de fréquence fondamentale du synthétiseur peut être réduit au delà du 200ms. La première base de données - « cn1 » de MBROLA pour la voix chantée et parlée de la langue chinoise a vu le jour.

5. DÉVELOPPEMENT DE L'APPLICATION

Nous avons conçu deux moyens différents pour contrôler le synthétiseur de voix chantée : la synchronisation du synthétiseur par un séquenceur MIDI pour l'utilisation en post-production ; ou bien une fonction qui nous permet de connecter directement le synthétiseur à un clavier MIDI pour une interprétation musicale en directe.

Quatre modules principaux sont inclus dans l'application (figure 2). Le module de synchronisation est développé pour recevoir les messages MIDI, il peut utiliser les messages MIDI pour piloter le séquenceur de la parole et la production de la voix. Le module de séquenceur de la parole est destiné à toutes les éditions de la parole, nous pouvons ajouter, modifier ou supprimer la parole dans une séquence, et la séquence de parole peut être enregistrée dans un fichier de texte après l'édition. Le module de règles gère les informations musicales, il peut convertir le format du message MIDI au format du synthétiseur MBROLA, par exemple la conversion de la hauteur de note ou encore les messages « note on/off ». Dans le module de la synthèse MBROLA, nous avons d'abord mis en place une bibliothèque de transcription phonétique, elle peut traduire la transcription PIN YIN en segments pour la base de données « cn1 ». Ensuite, nous avons développé un synthétiseur en temps réel basé sur l'objet « mbrola~ » pour la voix chantée chinoise.

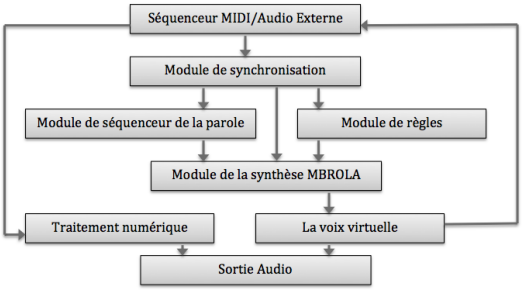


Figure 2. La structure de l'application

6. APPLICATION « MANDARIN REALTIME SINGING SYNTHESIS »

Cette application peut fonctionner en esclave en s'associant avec une autre application ou matériel séquenceur MIDI. Elle peut fonctionner de façon autonome pour une utilisation de l'interprétation *Live*.

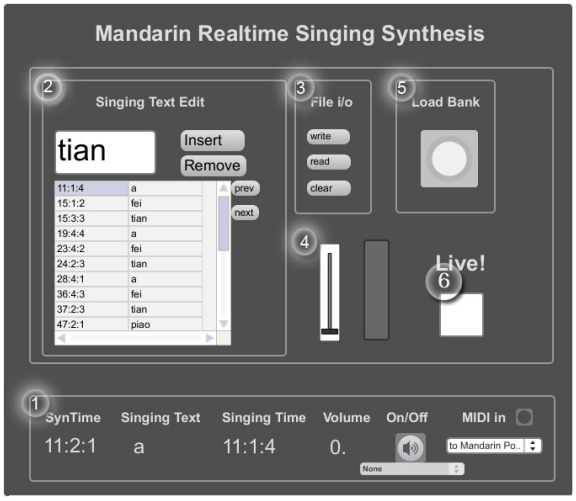


Figure 3. L'interface de l'application

L'interface d'application dispose de plusieurs modules (figure 3). En bas de la fenêtre, c'est le module de transmission ; nous voyons ici les informations du code de pointeur de Position, la parole en exécution, le code SMTPE, le volume, le menu pour la sortie audio, le bouton pour activer ou désactiver le synthétiseur MBROLA, le menu pour choisir le port MIDI, et un diode pour visualiser le signal d'entrée MIDI (figure 3, n°1).

Dans la partie en haut à gauche de l'interface, nous avons le module d'édition de la parole. Nous y trouverons le séquenceur de la parole, nous pouvons saisir, modifier ou supprimer la parole dans la séquence, les boutons « prev » « next » nous permettent d'avancer ou de reculer la position sur la séquence (figure 3, n°2).

Le module de « File i/o » est en haut dans le centre de l'interface, il inclut trois boutons – « write » « read » « clear ». Nous pouvons cliquer sur le bouton « write » pour sauvegarder la séquence de la parole dans un ficher texte externe. Nous pouvons charger une séquence de parole existante en cliquant le bouton « read ». Le bouton « clear » peut supprimer toutes les paroles dans la séquence actuelle (figure 3, n°3).

Au dessous du module « File i/o », nous avons un potentiomètre pour régler le volume général de la sortie audio, et à côté, nous avons un mètre de l'intensité du volume (figure 3, n°4).

En haut à droite de l'interface, nous avons le module « Load Bank », nous pouvons cliquer sur le bouton pour charger une base de données de MBROLA. Par exemple la base de données « cn1 » (figure 3, n°5).

Au dessous du module n°5, nous avons une case pour activer ou désactiver le mode *Live*. Une fois qu'il est activé, le synthétiseur va ignorer la synchronisation avec le séquenceur MIDI. La séquence de la parole s'enchaînera automatiquement après la réception de chaque message « note off » (figure 3, n°6).

Cette application est capable de produire la variation de hauteur de la voix linéaire. Par exemple, si nous produisons une syllabe /ma/ par une série de segments (figure 4), nous pouvons faire un *glissando* ou la voix *portamento* en modifiant la fréquence fondamentale de chaque segment comme la courbe rouge dans la figure 5. Au contraire, un changement non linéaire de la hauteur peut économiser le nombre de segment dans la production (veuillez-voir la courbe bleue dans la figure 5).

Figure 4. Une série de phonèmes enchaînés pour la prononciation /ma/

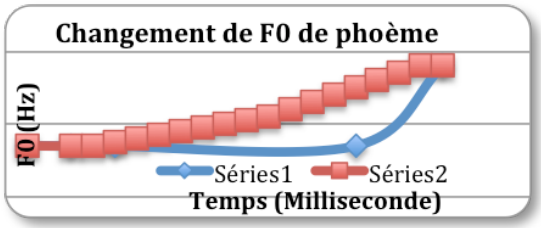


Figure 5. Changement de la fréquence fondamentale des phonèmes

Dans l'application « Mandarin Realtime Singing Synthesis », nous avons également intégré les fonctions de vibrato, et de *Pitch Bend*. Le synthétiseur peut répondre aux message de contrôle MIDI standard. Ces fonctionnalités peuvent améliorer la qualité de la voix chantée. La compatibilité de protocole MIDI nous

permet d'utiliser cette application avec de nombreux logiciels et matériels.

7. CONCLUSION

Dans cet article, nous avons fait une brève description du processus de développement de « Mandarin Realtime Singing Synthesis ». C'est le premier synthétiseur de la voix chantée basé sur une base de données diphonique en temps réel pour la langue chinoise. Cette application peut être utilisée dans le studio ou dans le concert. Les résultats obtenus de notre recherche pourront intéresser les musiciens et les chercheurs. De plus, La base de données « cn1 » que nous avons créée pendant le développement peut s'utiliser dans de nombreuses applications compatibles à MBROLA.

Néanmoins, notre recherche pourra être complétée et améliorée par des travaux ultérieurs, comme par exemples, l'amélioration de la latence du synthétiseur et la qualité de la production de la voix ; les possibilités de contrôle pour le traitement de la voix ; la diversification et spécification de la base de données en mandarin en créant des versions de voix d'enfant, ou masculine ; la réécriture de l'application par une langage de programmation de bas niveau ; la compatibilité de notre application avec les 34 autres langues de MBROLA, etc.

REFERENCES

[1] Cheng-Yuan Lin, J.-S. Roger Jang, Shaw-Hwa Hwang, "An On-the-Fly Mandarin Singing Voice Synthesis system", Proceedings of the Third IEEE Pacific Rim Conference on Multimedia: Advances in Multimedia Information Processing, Taiwan, 2003.

[2] Gu Hong Yan, Chen An Rui, "la méthode de la synthèse de la voix chantée en mandarin basée sur le modèle sinusoïde", Proceedings of the Conférence sur l'intelligence artificielle et les applications, Taiwan, 2004.

[3] Chowning John, "Frequency Modulation Synthesis of the Singing Voice", Proceedings of the Some Current Directions in Computer Music Research, MIT Press, Cambridge, UK, 1989.

[4] Johan Sundberg, "The KTH Synthesis of Singing", Proceedings of the Advances in Cognitive Psychology, Vol.2 No.2-3 p.133, Warsaw, Poland, 2006.

[5] Xavier Rodet, "Time domain formant-wave-function synthesis". pp.9-14, Computer music journal, 8(3), 1984.

[6] Cook Perry Raymond, "Spasm : a real-time vocal tract physical model editor/controller and singer : the companion software system", Proceedings of the Colloque sur les modèles physiques dans l'analyse, la production et la création sonore, Grenoble, France, 1990.

[7] Hideki Kenmochi, Hayato ohshita, "Vocaloid – Commercial singing synthesizer based on sample concatenation", Proceedings of the Centrer for Advenced Sound Technologies, Yamaha Corporation, Japan, 2007.

[8] T. Dutoit and H. Leich, "MBR-PSOLA : Text-to-Speech Synthesis Based on an MBE Resynthesis of the Segments Database", Proceedings of the Speech Communication, no°13, pp.435-440, Elsevier Publisher, USA, 1993.

[9] Nicolas D'Alessandro, Sebbe Raphaël, Bozkurt Baris, Dutoit Thierry, "Max/MSP Mbrola-Based Tool for reel-time voice synthesis", Proceedings of the EUSPIPC 05 Conference, Antalya ,Turkey, 2005.

[10] Louis Frécon, Okba Kazar, *Manuel d'intelligence artificielle*, p.554, PPUR Presses polytechniques, Lausanne, Suisse, 2009.

[11] Paul R. Frommer, Edward Finegan, *Looking at Languages: A Workbook in Elementary Linguistics*, p.365, Harcourt Brace College Publishers, San Diego, California, 1999.

A COMPUTER AIDED INTERPRETATION INTERFACE FOR JOHN CAGE'S NUMBER PIECE TWO[5]

Benny Sluchin
IRCAM/EIC
benny.sluchin@ircam.fr

Mikhail Malt
IRCAM/MINT
mikhail.malt@ircam.fr

ABSTRACT

Conceptual musical works that lead to a multitude of realizations are of special interest. One can't talk about a performance without taking into account the rules that lead to the existence of that version. After dealing with similar works of open form by Iannis Xenakis, Pierre Boulez and Karlheinz Stockhausen, the interest in John Cage's music is evident. His works are "so free" that one can play any part of the material; even a void set is welcomed. The freedom is maximal and still there are decisions to consider in order to make the piece played.

Cagener, is a project intended to develop a set of conceptual and software tools, that generates a representation of the work. It may be played live or showed as a sonorous installation. We deal here with the *Number Pieces* he composed in the last years of his life. The project calls for sound techniques, logic and musicological knowledge of the 21st century to approach the original ideas of the composer. The computer serves as a partner in making choices of multiple possibilities, mix together sounds of different sources and of various kinds and following compositional ideas clearly stated. The role of the sound projection in space is an important part of the studio preview.

1. INTRODUCTION

The performer approaching John Cage's music composed after the middle of the 20th century is often surprised to encounter a large amount of freedom mixed with a set of precise instructions. As a common result, the musician will determine "a version" in which he will decide on the free elements included in the score. A fixed score is thus created and used repeatedly. The performer will play it without any doubts of the composer's intentions. In fact, most of Cage's scores after the fifties are not to be pre-generated. Each performance should be unique and undetermined. Using the computer helps one to perform, ignoring what and when he is going to play.

2. SILENCE AND INDETERMINANCY IN JOHN CAGE'S EARLY PIECES

In connection with his encounter with Zen Buddhism [1], Cage rethinks his understanding of music. As a result, he composes *4'33"*, a work whose abandonment of intentional sound production drew controversy to his compositions. Cage spoke of silence in a new and positive way. Not only has it an importance in the creation of structure but one has to think of it not as an absence of sound but as a presence to fill an acoustical space.

2.1. The three kinds of "silence"

At first, Cage developed a structural concept of silence, considering it as an absence of sound helping to structure the music by its alternation with sound. The silence between the notes gave the work its cohesion. Later Cage adopted a spatial concept of silence, in which it was composed of all the ambient sounds that together formed a musical structure. Finally his concept evolved towards viewing silence as non-intention. Both sound and silence would exist only in the non-intention manner of nature [2].

2.2. What is indeterminacy?

The principal of indeterminacy allows the performers to work independently from each other. In this way, the musician ignoring the output of his fellow musicians will concentrate on his own part and the set of instructions, which imposes concentration even if degree of the freedom involved is high [3].

2.3. The *Number Pieces*

In Cage's *Number Pieces*, each individual part contains musical events endowed with time brackets, giving the player lower and upper bounds of time for starting and ending each event (**Figure 1**, **Figure 2**). The piece has a definite duration, and the elements occur within the given time brackets. In spite of the fact that only individual parts exist, an ensemble score is implicitly present and yields a strong form [5].

3. THE DIFFICULTY ON PERFORMING JOHN CAGE'S INDETERMINATE WORKS

It is the freedom relationship pre-determination that gives the player the main problem. Even if we find very hard instrumental passages, the main difficulties are: making the choice of when and what is to be played,

what order to choose for the elements, the amount of silence to insert between the events, and all this while ignoring the output of the other musicians involved. It has to be kept in mind that by the absence of intention, one should also ignore what he himself is about to perform. This means, that the entire score should be at the player's disposal, and that he will make up his mind intuitively and spontaneously. This research was initiated by that concern. We have approached first *Two*[5] and *Five*[3], as our initial concern pieces with trombone. The interface developed and the analytical results are easily applicable to any of the *Number Pieces*.

4. PERFORMING CONTEXT

4.1. Cage instructions

Cage's instructions in the TWO[5] are very brief. It consists of one line concerning microtonal notation for the trombone and one single paragraph concerning the general performance:

"Any changes of dynamics (pp and thereabouts for booth instruments) should be, like changes in breath, as imperceptible as possible. The piano should sound absent minded, without regularity of presence. If there is at some point a very short sound on the trombone it can be extremely loud, inexplicable" [15, score instructions]

4.2. Perfomance particularities

The "Number Pieces", in general, seem to be easy to perform, not presenting special instrumental difficulties. Concerning the way Cage's chooses the "material to fit in the time brackets" Benedict Weisser [14] point out the fact that:

"For as time passed, Cage was filling the boxes with progressively less and less."
[14, 94-95]

What Weisser call "boxes" is what we define as a "generic musical event". As Two[5] was composed in 1991, close to Cage's death, this statement gives some explanation concerning the scarcity of the material used. As there are very few elements every detail becomes important, as for example the attacks and releases of single notes, the simultaneity of chords, pedal actions that could bring unwanted noise or any other external sound source. The main difficulty consists in being "a tool" to make the composer's work comes to live. The musicians have to enter in a special state of mind where the awareness of quality of sound and the quality of silence are important. For this reason a "meditative concentration" is needed.

5. TOOLS FOR COMPUTER ASSSITED PERFORMANCE, CAP

5.1. What is "computer assisted performance"

The musical world offered itself a multitude of tools with the evolution of computer technologies. At first, dedicated to an employment in musical composition, they were oriented and adapted to a use in musical analysis and as aid tools to interpretation [4].

Several practices concerned with the interpretation field were developed. One can mention:

• The use of audio and MIDI sequencers as "super metronomes". It is common today that interpreters enter complete scores in sequencers as a way to work out difficulties in the performance (especially concerned with contemporary pieces). The musician can work progressively the problematic passages by varying the speed.

• The use of sequencers or notation programs to practice playing in ensemble. This is a logical extension of the "Minus-one" idea.

• The use of dedicated tools capable of correcting the player's interpretation.

An increasing number of composers prepare interpreters' oriented computer programs in order to help them play with the computer before starting with the actual musical piece.

There are other examples of computer tools created by or for interpreters, but our concern here is to show a new field developed in the last twenty years.

In our topic here, the interpretation of a category of Cage's work, in which the concepts of liberty and indetermination are predominant, it seems that the paper aspect of the scores is an obstacle in the realization. The wish that the interpreter could navigate freely, non-determined and without restraint through the musical material seems opposed to the fact that the music is presented on paper, and thus in a determined order. Computers may bring a solution to that particular difficulty for Cage's and also other composers' music. The actual playing prevents the musician from doing other tasks to orient his choices in "real-time". For example Iannis Xenakis in *Linaia Agon* (trio for horn, trombone and tuba, 1972) asks for a passage where the different instrumental choices are directed by a "gain matrix". The choice is computer-aided in order to enable a smooth interpretation [6]. *Duel* and *Strategy*, two other works by Xenakis based on Game Theory, received an analogue treatment for a CAP Interface [7]. Of different esthetics, *Domaines* de Pierre Boulez was investigated and lighted by an equivalent Interface [8].

5.2. From concepts to reality

How could one help the player, as well as possible, to perform the score in a context of "indetermination" and maximum of concentration? In what manner could one enable him to represent the Cage's musical thought?

Even if hard instrumental passages are apparently missing, the main obstacles to create a required musical atmosphere are: watching the chronometer, making the choice of when start and stop to play the musical events, the amount of silence to insert between the elements of the events, the quality of silence between events and all this while ignoring the output of the other musicians involved One possible solution was to provide an adapted interface. Here the choice is not only of timing but concerns the material itself.

5.3. Modeling musical pieces

One aspect of the tools proposed here is that they are oriented towards interpretation. In that concern, the interface should "contain" implicitly or explicitly all the instructions, constraints and concepts defined by the composer, as they will establish an "experimentation field". For the construction of CAP tools, the careful study of the pieces of John Cage and its formalization is necessary. The final interface will be, in a certain way, a computer model of the particular piece.

5.4. Modeling as a step in the musical analysis process

The construction of computer models of musical pieces is not a neutral process. It is fundamental to know well the works under study, understand the constraints left by the composer, as well as the historical context of its creation. But these are still insufficient in the modeling process. Every music work has a part of liberty and ambiguity. These "holes" has to be filled up to enable the modeling process. One has to take decisions as a function of his work assumptions, founded on musical and musicological bases. The necessity to represent the score or the processes suggested by the composer on numerical, symbolic or graphic spaces has great importance. Changing the representation of an object permits to see, to consider, to observe and finally to understand it, in a different manner. The modeling process is transformed in a pragmatic analysis of the musical phenomena [9].

6. THE "TIME-BRACKET" MODEL

There already exists an interface built for such performances [10] and a mathematical modeling of "time-brackets" [11] [12]. Our goal was to go beyond the interface as a score substitute, proposing to performers a tool to help them to find, at best, the "meditative concentration" needed (as explained in 4.2). But also, to try to build a model, from the scarce instructions left by John Cage, trying to fill the gaps with algorithms that could represent choice and indeterminacy, leading us to a better understanding of his composer craft. We have started to work with *Two*[5] (1991), a piece for trombone and piano, whose duration is 40 minutes. Music strips make up the individual parts (40 for the trombone and 29 for the piano). Each one is presented in the same way (**Figure 1**, **Figure 2**): a bold number indicating the strip order, and the "time-brackets" marked above (see 2.3).

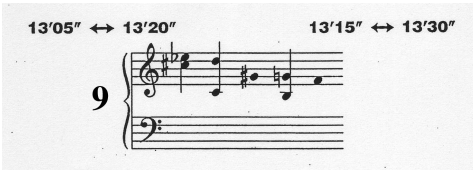


Figure 1: Piano 9th musical event

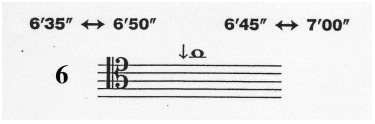


Figure 2: Trombone, 6th musical event

An important part of our reflection was to try to figure out an interface that could present these scores to the performers. This interface should help the player in the performance, and at the same time could help in generating studio previews.

We had implemented two models, an offline in "OpenMusic"[1] computer aided composition software, and a real-time one in MAX/MSP.

The first step in the process, was modeling a graphic representation of each "strip" as a musical event in time. For this, the time structure of the piece was represented as a set of events composed by the score time-line and a time vector. The time vector has the following structure: $\{strt_1, strt_2, end_1, end_2\}$, where $strt_1$ and $strt_2$ are the numbers in the left "time-bracket", and end_1 and end_2 the numbers in the right "time-bracket". I.e., $strt_1$ is the lower bound of the *Starting Time Zone* and $strt_2$ the higher one; end_1 is the lower bound of the *Ending Time Zone* and end_2 the higher one. The final graphic event had a trapezoidal shape (**Figure 3**), where the upper line represents the *Starting Time Zone* and the bottom line the *Ending Time Zone*. The height has no special meaning.

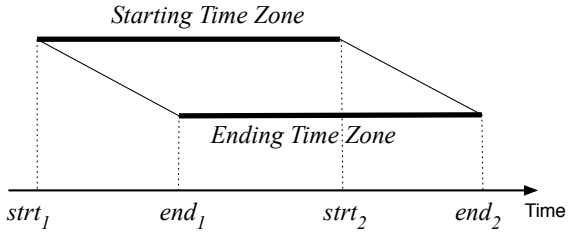


Figure 3: Trapezoidal graphic representation of each musical strip

More than being a graphic representation for each "strip", it allows us to identify similarities between

[1] "OpenMusic" is a software developed by Ircam by Gerard Assayag, Carlos Augusto Agon and Jean Bresson. See: http://recherche.ircam.fr/equipes/repmus/OpenMusic/.

generic musical events. For example, one can see easily the identity between the trombone generic musical events 4, 5, 14 and 26. The same comparaison done only on time brackets will be harder. We make a difference between a "generic musical event", and a "real musical event" A real musical event "i" is the one where the starting ($strt_i$) and ending (end_i) points are defined, i.e. a real musical event is a choice materilization. Where $strt_1 \leq strt_i \leq strt_2$ and $end_1 \leq end_i \leq end_2$. This one could be represented by a rectangle (**Figure 4**).

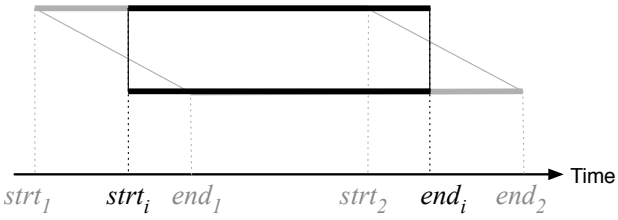


Figure 4: A "real musical event" represented as a rectangle

There are some properties one can easily be infered from the trapezoidal graphic representation, the generic event:
1. Cage's durations are $strt_2 - strt_1$ or $end_2 - end_1$, are a kind of nominal duration Cage gives to an event. The starting time span and ending time span are equal, resulting in a paralelogram, $strt_2 - strt_1 = end_2 - end_1$.
2. The maximum duration, $end_2 - strt_1$, is the maximum length an event can have.
3. The fact that , $strt_2 > end_1$ means that one can choose a starting point placed after the ending one, resulting in a void musical event. (idea so imporatnt to Cage, as he often indicates that the performer can choose, all, part, or nothing of the material to his disposal). In this case $strt_i > end_i$.
4. An implicit parameter that can be deduced is the "trapezoid slope", represented by the difference $end_1 - strt_1$ (as the heigth has no actual meaning). The slope is strongly connected with the performance. Concerning the trombone part, as it is wholy consisted of sustained notes, the knowledge of this parameter allows the performer to better manage his air capacity, in order to keep with the composer's indication. Regarding the pianist, the slope will be an information that allows him to manage his performance with regard to the time indications.

6.1. Offline model

The main purpose of the offline implementation was to study the possibility of generating several audio versions of the piece, and extracting parameters for musical analysis. For this we have built a first model in OpenMusic (**Figure 5**), with which we were able to:
1. Read text files with a representation of the time vectors (**Figure 6**),
2. Compute musical events with fixed "start and end" times,
3. Read audio recordings of each musical strip[2],
4. Rescale audio files to the durations computed in step 2,
5. Represent a *Two*[5] version (**Figure 7**) as graphical schema in a OpenMusic graphical interface (a Maquette), and
6. Save the data, derived from the calculation, in a file having the following data structure (**Figure 8**):
[*instrument starting_time duration sound_file*]

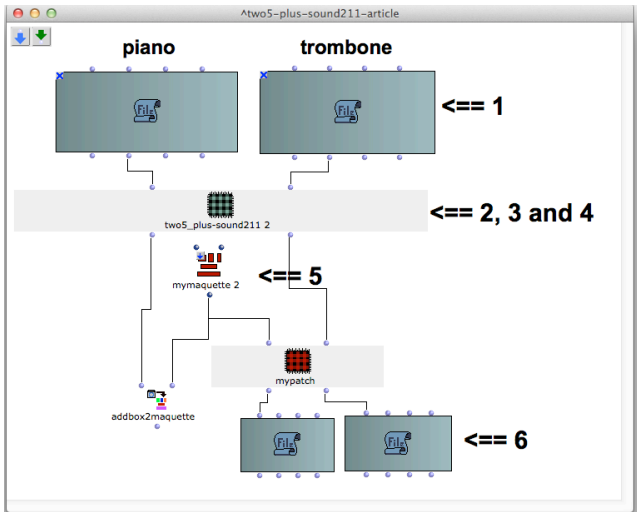


Figure 5: OpenMusic *Two*[5] calculations steps

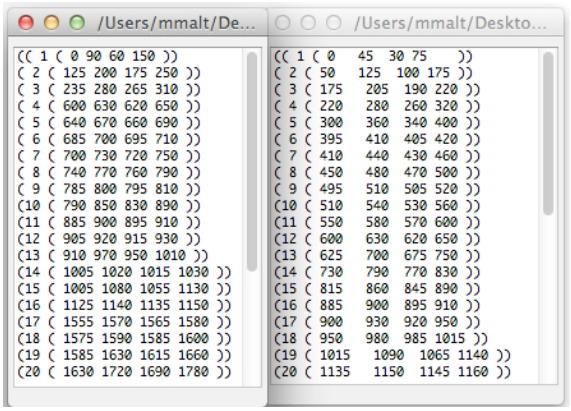


Figure 6: Piano and trombone, time vectors translated from Cage's "time-brackets"

This offline model, allowed us to, quickly, represent and evaluate a completely "indeterminate" performance.

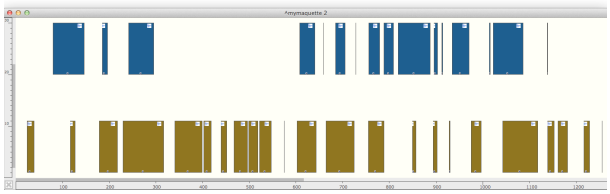


Figure 7: A *Two*[5] version (the first half of the piece with "real music events"), represented in "OpenMusic" maquette

[2] Benny Sluchin (trombone) and Sylvain Rapapport (piano) recorded this audio extracts.

```
(trombone   17000   36000   ts_trb_01_1.aif)
(piano      76000   11000   ts_pf_01_1.aif)
(trombone   95000   72000   ts_trb_02_2.aif)
(trombone   184000  15000   ts_trb_03_1.aif)
…
```

Figure 8: Extract of data from a calculated "version file" (time in milliseconds)

As a first algorithm we used a very single random process where for each event we calculated a fixed "*start_time*" and a fixed "*end_time*" as follow:

$$start\_time = \sigma(strt_1, strt_2) = strt_i$$
$$end\_time = \sigma(end_1, end_2) = end_i \quad (1)$$

Where,

$\{strt_1, strt_2, end_1, end_2\}$: are the elements from the time vector shown above, and $\sigma(a,b)$: is a random uniform function that chooses a value between (a,b). Naturally, other algorithms are under study. One should mention here the probabilistically approach of Alexandre Popoff [11] [12].

6.2. Real time model

The real time model had as main purpose, to offer an interface for the performance; it was built in the MAX/MSP[3] graphic programming environment.

The main interface (**Figure 9**), has 6 fields, some may be switched on or off according to the performer's wish.

6.2.1. *The global view – 1*

The global view displays a presentation of the entire duration of *Two*[5], using the trapezoidal event representation. It allows the performer to have a global view of the piece at a glance. As Cage mention about the context-specific character of his time-bracket notation:

"Then, we can foresee the nature of what will happen in the performance, but we can't have the details of the experience until we do have it."
[13, 182]

This global representation enables another perspective of the piece. The printed score orients a natural local view. For example, in this particular case, a five-part structure is easily perceived.

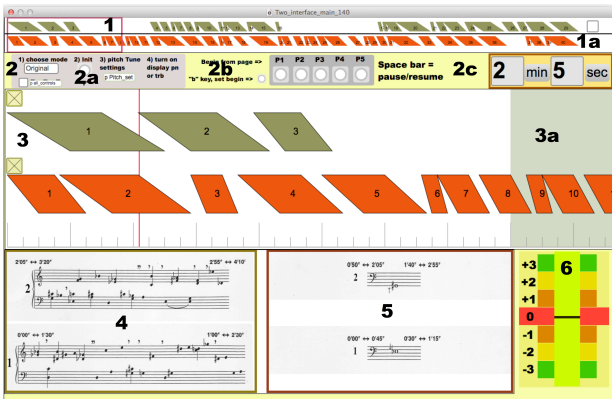


Figure 9. The main interface

6.2.2. *Main tool bar – 2*

Presents all the controls needed to calculate the events, start/stop the interface and a digital chronometer for the performance.

6.2.3. *Pitch tuning settings – 2a*

This zone allows setting the "seventh tone pitch tuner" (**Figure 11**) parameters: the input levels, the pitch analysis type (*fiddle~* by Miller Puckette, or *Yin*[4] by Chevigné&Kawahara), the smoothing analysis settings and the reference pitch.

6.2.4. *Page zone – 3*

The page zone is the main interface field. Here, the trapezoidal events (or, as in **Figure 10**, "real music events", showed as rectangles, in correspondence with the calculus and performance mode chosen) are displayed. A time cursor runs on a "page" whose horizontal size, in this case, is 8 minutes.

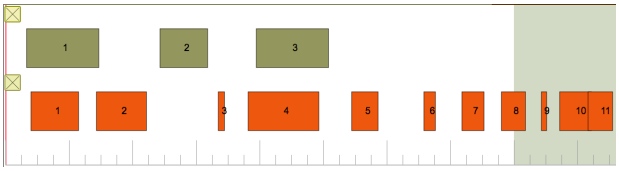


Figure 10. "Real music events" representation

The performer can choose to display, or not, the global view or the events (his own events or the partner events).

6.2.5. *Shadow view – 3a*

This sub field, allows the performer to anticipate, viewing the first 90 seconds of the next "page".

[3] © www.cycling74.com.

[4] The Yin algorithm was implemented in MAX/MSP by Norbert Schnell.

6.2.6. *Piano score strip field – 4*

This field displays the piano music strips from the original Cage score.

6.2.7. *Trombone score strip field – 5*

This field displays the trombone music strips from the original Cage score.

6.2.8. *Seventh tone tuner - 6*

This field is a display allowing the trombone player to tune, and check its tune (**Figure 11**). Cage asks for a particular microtonal setting, dividing a semitone in seven equal steps.

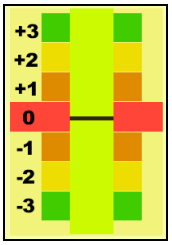


Figure 11. Seventh-tone tuner

The Cage's seventh tone concept is far from intuitive, bringing huge difficulties in the control of such demand. The performer has to practice in order to gain facility in this aim. This part of the interface would help to train the ear and get closer to the written score.

7. CONCLUSIONS

As stated earlier, the modeling process is transformed in a pragmatic analysis of the musical phenomena that leads us, step by step, to model some of Cage's concepts.

We have performed *Two*[5] several times using conventional setting (printed parts and stop-watches). A clear preference towards minimum visual communication and good acoustical space was stated.

The CAP interface, we presented, helped to get everything on the screen: the music to play, the timing, and a tuner for the microtonal control. The music was liberated from original paper pages representation and unveiling the form of the piece, hidden in its original form. Concerning the players, they can concentrate on performing when using a CAP interface. After determination of the "real musical events", in the context of "Number Pieces", they do not have to prepare any personal version, they do not need to be distracted by watching a chronometer, they do not need to loose their concentration handling paper pages or calculating start or ending times from the "time-brackets". One might wonder: when all decisions regarding the starting and ending points from events of the scores are made by a computer, what remains to be done by the performer/interpreter? The performers can ignore completely what music is being performed and just concentrate in their own performance. They can focus on the sound and silence quality required. The pianist can manage his time performance, in order that each musical event "fit" in the "real musical event" calculated, and the trombonist could be aware of his breath and tuning. In this way, the computer interface is a way to free the performers from interfering tasks, in order to get closer to Cage's original instructions.

Concerning future research:

1.Investigation of other Number Pieces, and even enlarging to other Time-brackets works, such as "Music for …" series (1984-7). Always taking into account the composer's instructions.

2.The interface we have developed can be easily adapted to other works. In cases of large numbers of performers, the zone giving information on the structure of the piece, could be used for musicological reasons, and switched off during a performance.

8. REFERENCES

[1] D. Nicholls, *John Cage*, University of Illinois Press, Unbana and Chacago, 2007.

[2] James G. Chilton, *Non-intentional performance practice in John Cage's solo for sliding trombone*, DMA dissertation, University of British Columbia, 2007.

[3] J. Pritchett, *The Music of John Cage*, Cambridge University Press, Cambridge, 1993.

[4] Mikhail Malt, Benny Sluchin, "L'interprétation assistée par ordinateur", *Le journal de l'Association des Anciens de Sciences et Musicologie*, Hors-série, n° 1, Avril 2011, p. 17-21.

[5] R. Haskins, *The Number Pieces of John Cage*. DMA dissertation, University of Rochester, 2004. Published as *Anarchic Societies of Sounds*, VDM Verlag, 2009.

[6] B. Sluchin, "*Linaia Agon*, towards an interpretation based on the theory" (Proceedings of Iannis Xenakis International Symposium, May 2005, Athens, Greece, p. 299–311.

[7] B. Sluchin, M. Malt, "Play and game in *Duel* and *Strategy*" *Xenakis International Symposium 2011*, London 1-3 April 2011.

[8] B. Sluchin, M. Malt, "Open form and two combinatorial musical models: The cases of *Domaines* and *Duel*", *MCM (Mathematics and Computing in Music) Proceedings*, Springer Verlag, 2011, p. 257–269.

[9] D. Keller D, B. Ferneyhough, "Analysis by modeling: Xenakis's ST/10 080262", *Journal of New Music Research*, vol. 33, Number 2, 2004, p. 161-171.

[10] Georg Hajdu, "Quintet.net: An Environment for Composing and Performing Music on the Internet", *Leonardo*, Volume 38, Number 1, February 2005, pp. 23-30.

[11] Alexandre Popoff, « Indeterminate Music and Probability Spaces: the Case of John Cage's Number Pieces », *MCM (Mathematics and Computing in*

Music) Proceedings, Springer Verlag, 2011, p. 220–229.

[12] Alexandre Popoff, « John Cage's Number Pieces: The Meta-Structure of Time- Brackets and the Notion of Time », Perspectives of New Music Volume 48, Number 1, Winter 2010, p. 65-83.

[13] John Cage, Joan Retallack, Musicage: *Cage muses on words, art, music*, Wesleyan university Press, 1996.

[14] Benedict Weisser, *Notational Practice in Contemporary Music: a Critique of Three Compositional Models (L. Berio, J. Cage, B. Ferneyrough)*. University of New York, Ph.D Dissertation, 1998.

[15] John Cage, *TWO[5], for piano and tenor trombone*, Peeters Editions, 1991.

DIGITARTIC : SYNTHÈSE GESTUELLE DE SYLLABES CHANTÉES

Lionel Feugère
LIMSI-CNRS et UPMC
lionel.feugere@limsi.fr

Christophe d'Alessandro
LIMSI-CNRS
cda@limsi.fr

RÉSUMÉ

Nous présentons le Digitartic, un instrument musical de synthèse vocale permettant le contrôle gestuel de l'articulation Voyelle-Consonne-Voyelle. Digitartic est situé dans la continuité de l'instrument de voyelles chantées synthétiques Cantor Digitalis, utilisant la synthèse par formants et développé dans l'environnement Max/MSP. Les analogies entre geste percussif et geste de constriction lors de la production de consonnes sont développées. Digitartic permet le contrôle temps réel de l'instant articulatoire, de l'évolution temporelle des formants, des bruits d'occlusion et de l'aspiration. Le lieu d'articulation peut varier continument par interpolation des lieux d'articulation des consonnes de référence. On discute du type de modèle de contrôle à utiliser suivant l'application recherchée, en s'appuyant sur des analogies gestuelles et des contraintes de temps réel. Un modèle de synthèse de l'articulation est présenté, utilisant les possibilités de contrôle d'une tablette graphique. Des exemples de syllabes synthétiques démontrent que le concept de contrôle gestuel de l'articulation, par analogie à la percussion, est valide.

1. INTRODUCTION

1.1. Propos de ce travail

Les voyelles de la parole correspondent à des conformations stables du conduit vocal, sur des durées qui peuvent atteindre plusieurs secondes en voix chantée (voire beaucoup plus dans certains styles de chant). Au contraire, les consonnes correspondent à des sons transitoires, relativement brefs, associés à des gestes de constriction totale ou partielle dans le conduit vocal. Les gestes consonantiques présentent ainsi des analogies avec les gestes d'attaque des sons musicaux, et en particulier les gestes de percussion digitale, ou de jeu des claviers manuels.

Dans la continuité de nos travaux sur le contrôle gestuel de la synthèse vocale, nous explorons dans cet article les analogies entre percussion manuelle et production de consonnes chantées. Par exemple, la trajectoire du geste de frappe sur une percussion est analogue à celle du geste de constriction des articulateurs sur eux-même. Le lieu et le mode d'articulation des consonnes ressemblent au lieu et au mode des frappes par exemple sur une percussion.

Ces analogies sont d'ailleurs utilisées depuis l'antiquité dans l'apprentissage de certains styles musicaux. Plus récemment, en Inde par exemple, les percussionnistes utilise un lexique de syllabes pour apprendre, mémoriser, transmettre et reproduire des séquences rythmiques, sur des instruments comme les tablas qui offrent plusieurs modes et lieux de frappe.

Une consonne n'est jamais produite seule, l'unité minimale de production de la parole étant la syllabe. La production de syllabes est un geste complexe, mettant en jeu la coordination des différents articulateurs de l'appareil vocal. Sur des durées brèves (de l'ordre d'une dizaine de millisecondes), les articulateurs (lèvres, langue, mâchoires ou luette) et la source vocale doivent se synchroniser, afin de changer dynamiquement la configuration du conduit vocal et de filtrer l'onde acoustique issue de la source glottique. La synchronisation entre les articulateurs et la vibration des plis vocaux permet de contrôler le voisement des sons produits.

Parmi les approches possibles pour synthétiser de la parole, c'est la synthèse à formant que nous avons explorée, plutôt que la synthèse par échantillonnage. Le modèle de contrôle des systèmes utilisant la synthèse par formants est constitué de règles pour la formation des unités phonologiques et des phrases ([15] pour une revue de littérature des systèmes de synthèse par formants). Nous proposons de transférer une partie de ces règles au niveau du geste de l'utilisateur. Toute la dynamique de la synthèse se retrouve alors dans le geste : la durée et l'évolution temporelle des transitions entre positions articulatoires, ainsi que le passage d'une syllabe à une autre sont contrôlés par l'utilisateur. Ce transfert du contrôle, en remplaçant des règles fixes par un geste humain, rapproche le mouvement des paramètres de synthèse du mouvement des articulateurs naturels. Les gestes manuels imitent les gestes articulatoires.

Il n'est pas proposé de produire toutes les unités phonologiques d'une langue. Il faut pour le moment restreindre la combinatoire des phonèmes. Le but n'est pas la synthèse d'un texte quelconque, mais la synthèse réaliste de syllabes, dans une perspective musicale. L'ensemble du travail est développé dans l'environnement Max/MSP [20].

1.2. Travaux antérieurs

La synthèse temps réel de l'articulation, c'est à dire le contrôle gestuel de l'articulation, a commencé avant l'arrivée de l'électricité, en 1791 avec la machine mécanique de Von Kempelen qui pouvait émettre une vingtaine de sons différents par un contrôle manuel reproduisant l'action des articulateurs [11]. Avec l'arrivée de l'élec-

tricité, Stewart inventa l'ancêtre des synthétiseurs à formants, composé d'une source périodique et de deux résonateurs électriques permettant de produire des voyelles, des diphtongues, et quelques mots tels que "mama, anna" [18]. En 1939, le premier synthétiseur capable de produire des phrases entières fût le VODER de Homer Dudley, modification du Vocoder mais avec des commandes manuelles [10]. En 1998, Fels et al. publient le GloveTalkII qui relie des mouvements de mains aux paramètres de contrôle d'un synthétiseur à formants, mouvements reconnus à l'aide de réseaux neuronaux. Les deux premiers formants des voyelles sont contrôlés par la position de la main gauche, les consonnes sont déclenchées par la main droite, exceptées les occlusives qui le sont par une pédale [12]. En 2000, Cook et al. présentent une interface de contrôle basée sur un accordéon [4] utilisant le système SPASM basé sur des guides d'onde numérique. Le souffle est contrôlé par le soufflet de l'accordéon, la hauteur tempérée par le clavier de l'accordéon. Une série de boutons a été ajoutée pour contrôler les voyelles et les consonnes [3]. D'Alessandro et al. mentionnent dans la publication du Handsketch l'utilisation de capteurs FSR pour déclencher des syllabes mais cet aspect n'est pas implémenté [6]. En 2011, Beller et al. utilisent des mouvements percussifs captés par un accéléromètre pour déclencher des séquences de voix préenregistrées [2]. Enfin, Astrinaki et al. ont présenté en 2011 un prototype de système temps réel de synthèse à partir de texte utilisant des modèles de Markov cachés (HMMs). Les HMMs modélisent les dépendances entre les phonèmes à partir d'une base de données de voix naturelle. Ces phonèmes sont concaténés et la modification de leur spectre, de l'intonation et des durées est réalisée à partir des HMMs [1].

La synthèse par formants permet facilement l'interpolation de valeurs de référence et offre ainsi un espace continu de positions articulatoires où l'on peut s'y déplacer sans restriction. D'où l'intérêt d'un contrôle gestuel du séquencement par règles et non par échantillons. Le principal inconvénient de la synthèse par formant est sa qualité sonore à priori plus faible que celle de séquences enregistrée. Mais nous pensons que les possibilités accrues de contrôle dynamique pallient le manque de qualité statique des sons.

Dans la suite de cet article, nous commencerons par présenter le modèle de production de l'instrument, de type source-filtre avec ses règles de trajectoires formantiques, de bruit d'occlusion et d'aspiration suivant les instants articulatoires, les lieux et modes d'articulation et la force vocale. Puis nous traiterons des modèles de contrôle de ce synthétiseur, de leur intérêt face aux systèmes existants, en discutant des gestes et interfaces appropriés pour le contrôle de l'articulation, et en comparant les trajectoires formantiques de la voix de synthèse et de la voix naturelle.

2. LE MODÈLE DE PRODUCTION DE DIGITARTIC

2.1. Le modèle de source CALM

Le modèle de source glottique utilisé dans ces travaux est le RT-CALM [5], variante temps réel du modèle CALM [8]. Ce modèle travaille dans le domaine spectral et s'appuie sur une analyse des principaux modèles temporels proposés dans la littérature [9].

Les propriétés spectrales de la source sont décrites par la forme de la dérivée de son spectre : un maximum en basses fréquences, le "formant glottique" et la pente spectrale pour les fréquences médium et aigües. L'onde de débit glottique (ODG) est souvent représentée par sa dérivée en prenant en compte le filtre passe-haut associé au rayonnement aux lèvres. Notons que le "formant glottique" ne correspond pas à un formant résonantiel mais à la forme de l'ODG. La force de voix est modélisée par une augmentation de l'intensité du signal et par la diminution de la pente spectrale de l'ODG dérivée.

2.2. Règles pour les transitions articulatoires

On se limite ici à la synthèse de syllabes de type VCV où V est une voyelle (/a/ pour les exemples) et C est soit une occlusive (/p,t,k/ comme référence), soit une semi-voyelle (/w,ɥ,j/ comme référence). Dans notre système, une transition articulatoire est définie par l'évolution temporelle des paramètres suivants :

- les valeurs des 4 premiers filtres formantiques (fréquence centrale, bande passante, amplitude)
- le bruit d'aspiration, modélisé par un bruit blanc modulé par la forme de l'ODG.
- les coefficients de filtres modifiant un bruit blanc pour le bruit occlusif
- la force vocale

2.2.1. Transitions formantiques et d'aspiration

On utilise les valeurs cibles des consonnes et de la voyelle /a/ qu'on interpole pour obtenir des positions articulatoires intermédiaires à un instant temporel défini par le contrôle gestuel. On choisit une interpolation linéaire pour la correspondance entre les valeurs des formants et le paramètre de contrôle de l'instant articulatoire, et on laisse l'utilisateur modifier cette linéarité par la dynamique de son geste associé au paramètre de contrôle de l'instant articulatoire.

2.2.2. Bruits d'occlusion

On modélise les bruits d'occlusion à l'aide d'un bruit blanc filtré par une série de filtres en cascade, dont on détermine la forme à l'aide du spectre de bruit de voix réelles.

Afin de modéliser l'influence de la voyelle, on fait l'hypothèse que le bruit consonantique sera d'autant plus marqué par les résonances formantiques que son lieu de constriction est postérieur. Le bruit est filtré par les filtres for-

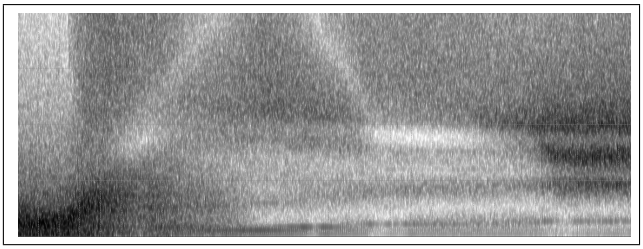


Figure 1. Sonogramme ($0 - 8000$ Hz) du bruit d'occlusion produit par le Digitartic, avec le lieu d'articulation qui évolue le long de l'axe bilabial - alvéo-dental - palatal (de gauche à droite)

mantiques correspondant à la voyelle, avec une bande passante d'autant plus étroite que le lieu d'occlusion est postérieur. Par exemple, un bruit labial est peu modifié car la source du bruit est situé à l'extrémité du conduit vocal, et un bruit provenant de la glotte est filtré comme l'est une voyelle. Cette caractéristique est illustrée par le sonogramme de la figure 1 où l'on remarque l'apparition progressive des formants quand le lieu d'origine du bruit devient postérieur.

2.3. Continuité du lieu d'articulation

Le système permet de réaliser des consonnes à des lieux intermédiaires entre deux consonnes de référence d'un même mode d'articulation, et également de contrôler l'instant précis d'articulation. Le paramètre correspondant au lieu d'articulation étant continu, on peut alors produire une infinité de pseudo-consonnes (consonnes qui n'entrent pas dans le système phonologique de la langue visée).

A partir des valeurs des fréquence / amplitude / bande-passante de leurs formants, et des valeurs des coefficients des filtres des bruits consonantiques, on construit des pseudo-consonnes intermédiaires sur l'axe bilabial - alvéo - dental - palatal. L'hypothèse utilisée revient à supposer qu'on peut interpoler ces valeurs pour obtenir des niveaux d'articulation intermédiaires. C'est peut être le cas en première approximation entre les occlusives alvéo-dentales et palatales, mais plus difficilement concevable entre les occlusives bilabiales et alvéo-dentale, vu la discontinuité entre ces deux lieux d'articulation. Du point de vue de la synthèse de voix pour la musique, cela permet d'obtenir des sons vocalement plausibles, bien que difficilement prononçable en réalité. Cependant, la perception des consonnes étant catégorielle (on associe la consonne entendue au plus proche phonème de notre langue), la perception qu'on a de cette continuité consonantique est discrète en ce qui concerne l'identification du son. La continuité des consonnes s'exprime par un changement progressif de la qualité perçue de la consonne, mais en général pas de son identification.

La figure 2 présente une série d'occlusives de l'instrument (suivies chacune de la voyelle /a/) pour lesquelles le lieu d'articulation évolue progressivement de bilabial à palatal. La configuration passe donc par les syllabes /pa/, /ta/ et /ka/. De la même manière, la figure 3 est une série de semi-voyelles pour lesquelles le lieu d'articulation varie, en passant donc par /wa/, /ɥa/ et /ja/.

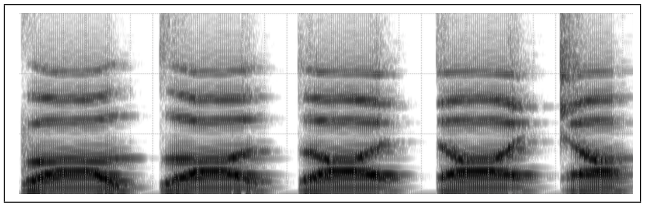


Figure 2. Sonogramme ($0 - 6000$ Hz sur 3 secondes) de "Occlusive-/a/" successifs produits par le Digitartic, avec le lieu d'articulation de l'occlusive qui évolue sur l'axe bilabial - alvéo-dental - palatal

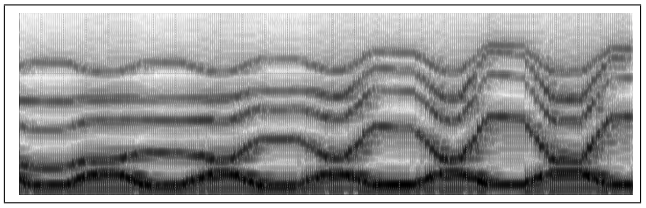


Figure 3. Sonogramme ($0 - 6000$ Hz sur 3 secondes) de "SemiVoyelle-/a/" successifs produits par le Digitartic, avec le lieu d'articulation de l'occlusive qui évolue sur l'axe bilabial - alvéo-dental - palatal

2.4. Continuité du mode d'articulation

La synthèse est dans cet article restreinte à deux modes d'articulation : les semi-voyelles et les occlusives. La différence fondamentale dans la manière de produire les consonnes de ces deux modes d'articulation est l'ajout d'un bruit blanc filtré au système source-filtre. Le reste des différences portent sur des règles distinctes, mais des processus identiques (changement des formants, du bruit d'aspiration, de la force vocale), et sur les gestes de l'utilisateur.

Dans le présent système, on considère les modes d'articulation comme discrets, c'est à dire qu'on passe des occlusives aux semi-voyelles sans intermédiaire possible. Cependant, les occlusives d'un lieu d'articulation donné présentant à peu près les mêmes valeurs formantiques que les semi-voyelles du même lieu d'articulation, on peut envisager facilement de créer une continuité entre ces deux modes d'articulation à l'aide de règles sur l'intensité des bruits (d'inexistant pour les semi-voyelles à fort pour les occlusives), de la force de voix (pleine pour les semi-voyelles tenues et nulle pour les occlusives sourdes), et de gestes de vitesse différente.

3. LE MODÈLE DE CONTRÔLE DE DIGITARTIC

On cherche à externaliser les mouvements internes de l'appareil vocal (larynx, uvule, langue, mâchoire, lèvres) dans leur ensemble par des gestes manuels. Il faut donc rechercher des interfaces qui permettent des gestes adaptés au contrôle articulatoire. Le modèle de contrôle est ce qui

permet de relier les paramètres du modèle de production à l'interface de contrôle.

3.1. Quelle interface pour quelle application ?

La multiplicité des paramètres à contrôler dans un synthétiseur vocal implique que pour un résultat optimal, on se doit de choisir une interface vraiment adaptée à l'application recherchée. Pour une approche musicale, on peut dans un premier temps ne pas chercher à reproduire tous les phonèmes de la langue étudiée. Les expériences passées ont montré que les instruments de synthèse de voix parlée sont très difficile à contrôler. Les opérateurs du VODER [10] devaient être entrainés pendant au moins un an avant de pouvoir synthétiser des phrases en public. Plus récemment, le Glove-talkII [12], qui relie les mouvements des mains à un synthétiseur par formant, semble nécessiter beaucoup d'heures de pratique avant de pouvoir parler de façon à peu près intelligible, même si les derniers environnements utilisant le Glove-TalkII permettent un apprentissage plus rapide [19] [13].

Parmi les systèmes cités en introduction, seul le Glove-TalkII permet de contrôler l'instant d'articulation et de choisir a priori des lieux d'articulation autres que ceux de la langue choisie pour leur système. Mais dans une perspective musicale nécessitant une haute précision temporelle, l'utilisation de gants haptiques présente l'inconvénient d'une latence importante, de l'ordre de 10 à 20 ms comme le mentionne Kunikoshi et al. dans son récent système de synthèse temps réel destiné à la communication pour personnes muettes [16].

Dans Cantor Digitalis [14], le contrôle précis porte sur la mélodie (F_0). La tablette graphique munie de son stylet, est contrôlée par un geste proche de l'écriture. Or l'écriture requiert une haute précision spatiale : la tablette graphique est très bien adaptée au contrôle de F_0 pour la musique, qui nécessite elle aussi une haute précision de l'ordre de 4 centièmes de demi-tons.

Ici, on s'intéresse au contrôle des syllabes et leurs articulations, comme par exemple les récitations onomatopéiques des percussions indiennes. Le contrôle de F_0 est alors réduit à un contrôle ne nécessitant pas une grande précision. Mais s'il on veut "scatter", c'est à dire chanter en utilisant des onomatopées, alors le contrôle de F_0 devra être précis. L'interface devra au moins inclure les contrôles suivants de :

- F_0 et du lieu d'articulation (continus et précis)
- l'instant articulatoire (continu et rapide)
- la force vocale (continu)
- le mode d'articulation (discret)
- les voyelles (continu ou discret)

La tablette graphique permet un nombre important de contrôles (pression, position X/Y, orientations du stylet) mais le stylet ne doit pas quitter la tablette. L'analogie entre geste percussif et geste articulatoire nous encourage à contrôler l'articulation à l'aide de mouvements verticaux de type frappe. Les mouvements de la main ou des doigts sont facilement mesurables à l'aide d'accéléromètres. Des tests de mapping sont en cours.

3.2. Geste percussif et geste articulatoire

Le geste articulatoire et le geste percussif présentent des analogies assez fortes entre :

- le lieu d'articulation et le lieu de frappe
- le mode d'articulation et la manière de frapper une percussion
- le caractère voisé d'une consonne et le caractère ouvert/fermé d'une frappe (i.e main laissée ou non en contact avec la peau après la frappe pour éviter sa vibration)
- les mouvements de coarticulation et les mouvements entre deux frappes successives

En revanche, nous n'affirmons pas qu'il existe une ressemblance psycho-motrice stricte des deux types de gestes.

A l'aide d'une interface multitouch trackpad, on peut facilement modéliser une peau de percussion et faire correspondre les analogies présentées plus haut. Le problème est le contrôle de l'instant articulatoire qui se fait par analogie avec la position verticale de la main qui frappe la peau. Or, le trackpad ne permet pas de capter le geste vertical. Pour remédier à cela, un modèle de contrôle a été établi à l'aide d'un trackpad qui déclenche la transition VC lors de la pose du doigt et déclenche la transition CV lors du retrait du doigt. Le contrôle sur l'instant articulatoire est donc réduit au déclenchement des différentes transitions, mais sans correspondance continue avec le geste. Ici, le trackpad contrôle les aspects consonantiques (lieu et mode d'articulation, déclenchement des phases d'articulation) avec la main secondaires et une tablette graphique permet de contrôler les aspects vocaliques (mélodie, force vocale) avec la main préférée.

Dans l'idéal, nous aimerions avoir une captation continue de la dynamique du doigt pour le faire correspondre en temps réel avec l'instant articulatoire. On peut s'interroger alors sur les correspondances entre phase articulatoire (VC versus CV) et phase de geste percussif (remontée et descente du geste). Est-il préférable de faire correspondre la phase de frappe (descente) du doigt avec la transition CV et la phase de remontée du doigt avec la transition VC ou bien le contraire ? Cela nous amène à devoir introduire dans la discussion le temps réel et la notion d'attaque en musique.

3.3. Temps réel et contrôle articulatoire

Un instrument de synthèse vocale qui ne contrôlerait que le déclenchement des phases articulatoires et non le contrôle de l'instant articulatoire sont voués à ne pas pouvoir être joués dans des musiques nécessitant une haute précision temporelle, comparable au seuil de perception de la non simultanéité de deux évènements, de l'ordre de la dizaine de millisecondes.

En effet, hormis les occlusives pour lesquelles l'attaque musicale (au sens de son "centre perceptif", ou P-center) se situe au début de la transition CV en même temps que l'explosion, l'attaque musicale d'une syllabe se situe en fin de transition CV qui a une longueur de l'ordre de plusieurs dizaines de millisecondes (30 à 50 ms chez les semi-

voyelles ou liquides). De plus, à ce retard lié au modèle, s'ajoute le retard de la récupération des données de l'interface. En percussion, l'attaque se situe au contact de la main/doigt sur la peau. Dans l'idéal, il faudrait donc relier le mouvement descendant du geste manuel au resserrement des articulateurs pour les occlusives et au relâchement des articulateurs pour les semi-voyelles (de même pour les fricatives).

L'intérêt du trackpad vis à vis de la tablette est sa capacité à capter plusieurs doigts simultanément. Mais divers problèmes ont été rencontrés avec le trackpad qui nous ont amené à ne pas l'utiliser pour le profit de la tablette graphique seule :

- latence d'autant plus grande que le nombre de doigts en contact est important, allant jusqu'à des latences perceptibles de l'ordre de 100 ms ;
- bugs dans l'external Max/MSP qui récupère les données du trackpad ("fingerpinger" [1])

Ainsi, pour le moment, le geste percussif se fait dans le plan horizontal d'une tablette graphique. Capter le mouvement vertical de la main ou du doigt en temps réel n'est pas sans poser de problèmes techniques. Les mouvements sont d'amplitude assez faibles (moins d'une dizaine de 10 cm) et la résolution doit être suffisamment importante pour que le pas d'échantillonnage ne soit pas audible. Nous avons pensé à des systèmes de captation vidéo ou de gants haptiques, mais le temps de calcul ou la précision est mauvaise (jusqu'à plusieurs dizaines de millisecondes). La meilleure solution trouvée, à la fois sur la facilité d'utilisation et de la rapidité de la réponse, seraient des accéléromètres placées sur les doigts. La vitesse est récupérable par intégration. La position par double intégration fournit trop d'erreurs de calcul. Il faudrait donc faire correspondre la vitesse à l'instant articulatoire.

4. RÉSULTATS

Un schéma général de l'instrument utilisant une tablette graphique avec la deuxième configuration (explicitée ci-après) est donné à la figure 4. La première configuration est définit par les correspondances suivantes :

- pression du stylet sur la tablette $\longleftrightarrow$ force vocale
- position du stylet sur l'axe Y $\longleftrightarrow$ lieu d'articulation
- angle entre stylet et axe X $\longleftrightarrow$ instant articulatoire
- position du stylet sur l'axe X $\longleftrightarrow$ F_0

Dans la deuxième configuration, les contrôles du lieu d'articulation et de l'instant articulatoire sont inversés afin de pouvoir profiter de la bonne résolution spatiale et temporelle de la captation de la position du stylet :

- angle entre stylet et axe X $\longleftrightarrow$ F_0
- position du stylet sur l'axe X $\longleftrightarrow$ instant articulatoire

Alors que la première configuration permet de chanter à une hauteur mélodique précise, la production de syllabe est facilité et de meilleure qualité avec la deuxième configuration car la résolution de la position du stylet est plus

1 . www.anyma.ch/2009/research/multitouch-external-for-maxmsp/, consultée le 13 avril 2012

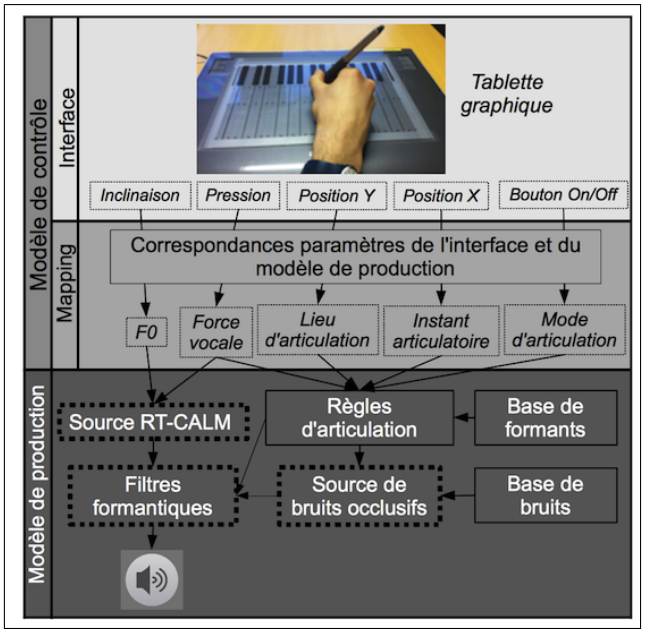


Figure 4. Shéma de fonctionnement général du Digitartic

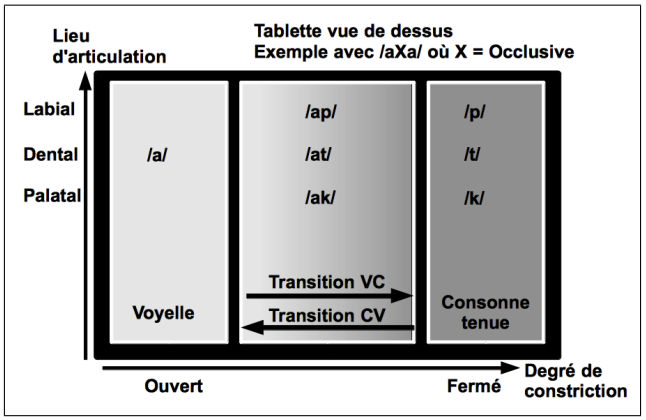


Figure 5. Tablette vue de dessus avec son mapping spatial pour le contrôle de l'instant articulatoire suivant l'axe X et du lieu d'articulation suivant l'axe Y

grande que celle de l'angle entre le stylet et l'axe X, et que le geste est plus simple (déplacement rectiligne de la main versus torsion du poignée tout en maintenant le stylet en contact fixe). Le contrôle de l'instant articulatoire et du lieu d'articulation dans l'espace 2D de la tablette, comme il est défini dans le deuxième configuration, est illustré par la figure 5.

Nous comparons ci-dessous quelques séquences VCV produites via la deuxième configuration avec les mêmes séquences VCV produite par une voix naturelle (figure 6 pour les semi-voyelles et figure 7 pour les occlusives).

Ces images appellent les remarques suivantes :

- la dynamique des trajectoires est bien reproduite
- les amplitudes des formants F_3 à F_5 des consonnes sont de façon générale trop élevées par rapport à ce qu'on peut observer en voix naturelle (figure 6).
- l'aspiration sur la transition VC dure trop longtemps pour les occlusives synthétiques comparées à la voix

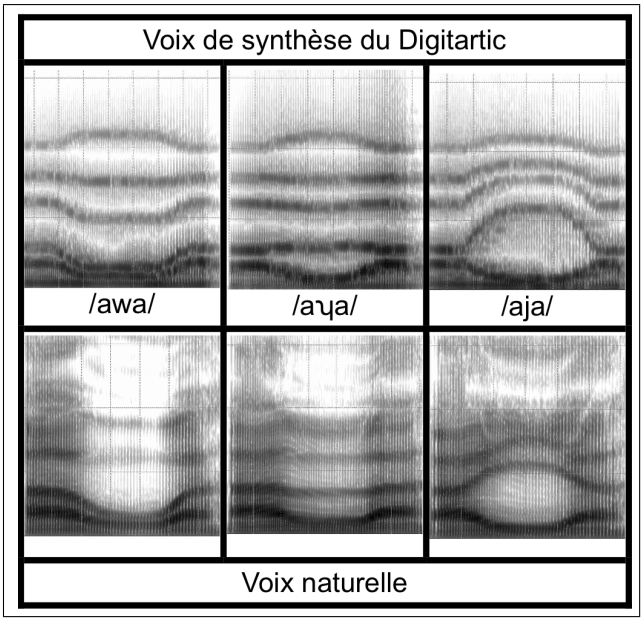


Figure 6. Sonogrammes (0-6000 Hz) des semi-voyelles de Digitartic et de voix naturelle

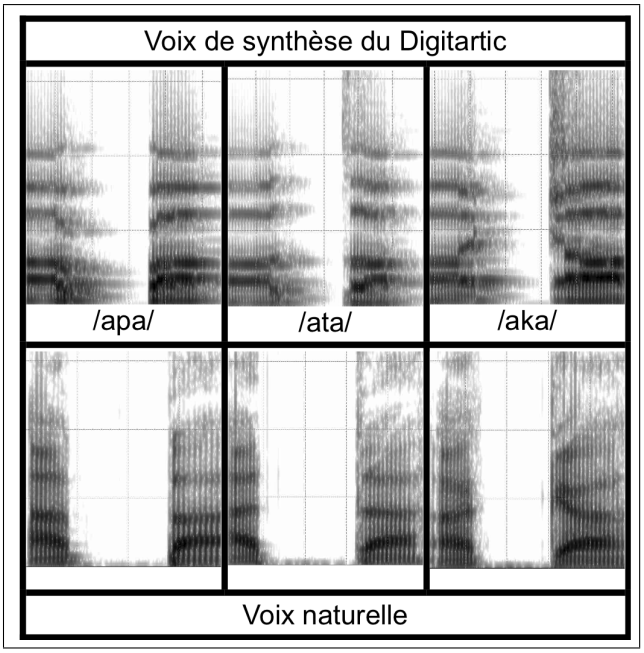


Figure 7. Sonogrammes (0-6000 Hz) des occlusives de Digitartic et de voix naturelle

naturelle (figure 7).

– les transitions avec le Digitartic présentent de petites discontinuités contrairement à la voix naturelle visible en zoomant sur les figures 6 et 7. Les raisons peuvent être : le geste rapide qui est exigé et la résolution spatiale et/ou temporelle limitée ne permettent peut être pas d'avoir suffisamment de données reçues de la tablette pour accomplir une trajectoires consonantiques bien continue puisqu'on a un contrôle directe sur la trajectoire de l'ensemble des formants ; Digitartic possède un module qui atténue en temps réel l'amplitudes des formants quand les harmoniques de F_0 et les filtres formantiques coïncident [14], pouvant créer des variations d'amplitudes très brèves quand les formants évoluent rapidement.

5. PERSPECTIVES

Le système Digitartic permet de démontrer qu'il est possible de contrôler finement et précisément par le geste manuel des transitions consonantiques convaincantes. L'analogie entre geste de constriction dans l'appareil vocal et geste de percussion manuelle semble donc prometteuse.

Nous avons pour ambition d'élargir les possibilités articulatoires du Digitartic aux autres modes d'articulation, comme les fricatives et les nasales. Le point de mire est toujours la performance temps réel et nous sommes en train de tester l'utilisation d'accéléromètres placés sur les doigts comme interface pour le contrôle de l'instant articulatoire et de l'intensité consonantique.

La synthèse d'articulation est introduite progressivement dans notre chorale, Chorus Digitalis [14] [17], un groupe musical qui se réunit de façon hebdomadaire pour faire de la musique à l'aide de synthèse vocale à contrôle gestuel. Quelques consonnes enrichissent déjà la palette sonore des chanteuses et chanteurs virtuels, ainsi que plusieurs modules pour améliorer la qualité de la synthèse par formants, dont des interactions source-filtre, et la modélisation de modulations naturelles de la source.

Enfin, notre instrument devra être validé par des expériences perceptives pour évaluer la qualité des consonnes produites et d'autres part par des expériences plus quantitatives pour mesurer les capacités de notre système à reproduire les transitions articulatoires.

6. REMERCIEMENTS

Les auteurs tiennent à remercier les relecteurs pour leurs bons conseils.

Ce travail est mené dans le cadre du projet Européen FEDER OrJo [2] dans lequel le LIMSI fournit des instruments vocaux pour le logicel Méta-Malette [7] qui permet de jouer des instruments virtuels audio-visuels à plusieurs.

7. REFERENCES

[1] Astrinaki, M., Babacan, O., d'Alessandro, N., Dutoit, T., *sHTS : a streaming architecture for statistical parametric speech synthesis*, International Workshop on Performative Speech and Singing Synthesis, Vancouver, BC, CA, March 14-15, 2011.

[2] Beller, G., *Gestural Control of Real-Time Concatenative Synthesis in Luna Park*, P3S 2011, International Workshop on Performative Speech and Singing Synthesis, Vancouver, BC, CA, March 14-15, 2011.

2 . Voir le site du projet OrJo http ://pucemuse.com/orjo

[3] Cook., P. R., *Identification of control parameters in an articulatory vocal tract model, with applications to the synthesis of singing*, PhD thesis, Stanford University, 1991.

[4] Cook, P., Leider, C., *SqueezeVox : A New Controller for Vocal Synthesis Models*, Proceedings of the ICMC (International Computer Music Conference), Berlin, 2000.

[5] D'Alessandro, N., d'Alessandro, C., Le Beux, S., Doval, B. *Real-time CALM Synthesizer New Approaches in Hands-Controlled Voice Synthesis*, Proceedings of the 2006 International Conference on New Interfaces for Musical Expression (NIME06), Paris, France, pp. 266-271, 2006.

[6] D'Alessandro, N., Dutoit, T. *HandSketch Bi-Manual Controller, Investigation on Expressive Control Issues of an Augmented Tablet* , Proceedings of the 2007 Conference on New Interfaces for Musical Expression (NIME07), New York, NY, USA, pp. 78-81, 2007.

[7] De Laubier, S., Goudard., V., *Puce Muse - La Méta-Mallette*, Journée d'Informatique Musicale, 2008.

[8] Doval, B., d'Alessandro, C., Henrich, N., *The voice source as a causal / anticausal linear filter*. In ISCA, editor, Proceedings of Voqual'03, *Voice Quality : Functions, analysis and synthesis*, Geneva, Switzerland, 2003.

[9] Doval, B., d'Alessandro, C., Henrich, N. , *The spectrum of glottal flow models*. Acta Acustica, 92 :1026–1046, 2006.

[10] Dudley, H. *Remaking speech*, The Journal of the Acoustical Society of America, 11(2) :169-177, 1939.

[11] Dudley, H., Tarnoczy, T. H., *The speaking machine of Wolfgang von Kempelen*, J. Acoust. Soc. Am., vol. 22, no. 2, pp. 151-166, 1950.

[12] Fels, S., Hinton, G., *Glove-TalkII : A neural network interface which maps gestures to parallel formant speech synthesizer controls*, IEEE Transactions on Neural Networks, pp. 205–212, Vol 9, No. 1, 1998.

[13] Fels, S., Pritchard, R., Lenters, A., *ForTouch : A Wearable Digital Ventriloquized Actor*, Proceedings of the International Conference on New Interfaces for Musical Expression, pp. 274–275, 2009.

[14] Feugère, L., Le Beux, S., d'Alessandro, C., *Chorus Digitalis : Polyphonic gestural singing*. P3S 2011, International Workshop on Performative Speech and Singing Synthesis, Vancouver, BC, CA, March 14-15, 2011

[15] Klatt, D. H., *Review of text-to-speech conversion for English*, J. Acoust. Soc. Am., Volume 82, Issue 3, pp. 737-793, 1987

[16] Kunikoshi, A., Qiao, Y., Saito, D., Minematsu, N., Hirose, K., *Gesture Design of Hand-to-Speech Converter Derived from Speech-to-Hand Converter Based on Probabilistic Integration Model*, Proceedings of Interspeech, pp. 3025-3028, 2011.

[17] Le Beux, S., Feugère, L., d'Alessandro, C., *Chorus Digitalis : experiments in chironomic choir singing*. Proceedings of Interspeech 2011, 28-31, Florence, Italie, August 2011.

[18] Lienard, J.-S., *Les processus de la communication parlée*, Masson, Paris, 1977.

[19] Pritchard, B., Fels, S., *GRASSP : Gesturally-Realized Audio, Speech and Song Performance*, Proceedings of the International Conference on New Interfaces for Musical Expression, pp. 272–276, 2006.

[20] Puckette, M., Zicarelli, D., *Max/MSP*, Cycling 74/IRCAM, version 5.1, 1990-2010.

ORJO ET LA META-MALLETTE 4.0

Serge de Laubier
Guillaume Bertrand

PUCE MUSE
www.pucemuse.com

Hugues Genevois
Vincent Goudard
Boris Doval

Institut Jean le Rond
d'Alembert (équipe LAM)
www.dalembert.upmc.fr/lam

Lionel Feugère
Sylvain Le Beux
Christophe d'Alessandro

LIMSI-CNRS
www.limsi.fr

RÉSUMÉ

Cet article décrit le projet de recherche OrJo 2009-2012 (Orchestre de Joysticks) qui associe quatre structures, PUCE MUSE, le LAM (UPMC), le LIMSI (CNRS, associé à l'UPMC et à l'Université Paris-Sud), et 3Dlized, autour de quatre grands objectifs :

1. réaliser quatre versions du logiciel plateforme pour s'adapter aux différents usages,
2. proposer une collection d'instruments virtuels sonores et visuels,
3. améliorer la représentation graphique des instruments virtuels,
4. pratiquer, échanger, et conserver un répertoire sur la Méta-Librairie.

Ce projet interroge plusieurs usages nouveaux comme la pratique en orchestre d'instruments virtuels, l'échange et la transmission de partitions interactives pour ces orchestres, l'apport du relief pour la musique visuelle. Il évoque aussi les développements d'instruments virtuels comme les instruments de synthèse vocale du LIMSI et les instruments par modèles physiques, modèles topologiques et modèles statistiques de l'équipe Lutheries acoustique musique (LAM).

Cet article est aussi une invitation à utiliser la plateforme Méta-Mallette [12] via son SDK (libre) et le site d'échanges Méta-Librairie.

1. INTRODUCTION

Depuis 2003, PUCE MUSE développe et expérimente un logiciel, la « Méta-Mallette », qui permet de jouer collectivement musique et image sur ordinateur en utilisant des interfaces gestuelles. Les premières expérimentations ont révélé un très vif intérêt des utilisateurs de tous âges et de divers horizons. En effet, la Méta-Mallette est un dispositif qui permet d'aborder de manière ludique et active des notions musicales et acoustiques. Elle favorise une transversalité des pratiques pouvant parfois associer artistes, professeurs de musique, d'arts plastiques, de technologies, de physique, de formation musicale (solfège), de musique électroacoustique ou de MAO, d'improvisation, de multimédia… La Méta-Mallette est ainsi utilisée par des musiciens, scientifiques, danseurs, plasticiens, et pratiquée dans les écoles maternelles et primaires, collèges, universités, mais aussi auprès du troisième âge, handicapés, amateurs ou professionnels. Depuis 2005, plusieurs améliorations du logiciel ont été réalisées en interaction permanente avec des pédagogues et des artistes.

OrJo (Orchestre de Joysticks) prolonge cette expérience en associant quatre structures sur ce projet: PUCE MUSE qui coordonne, le LAM (UPMC), le LIMSI (Paris 11), et 3Dlized société de production 3D relief.
Cet article détail les quatre grands objectifs cités ci-dessus, qui structurent ce projet jusqu'à fin 2012.

2. RÉALISER 4 VERSIONS DU LOGICIEL PLATEFORME POUR S'ADAPTER AUX DIFFÉRENTS USAGES

2.1. La Méta-Mallette

La version complète s'adresse aux « aventuriers », prêts à passer du temps pour élaborer des projets artistiques. Ils maitrisent les principales fonctionnalités du logiciel :

1. constituer des orchestres d'instruments virtuels audiovisuels (jusqu'à 30 instruments simultanés)
2. définir des « mapping » entre interfaces gestuelles et paramètres audiovisuels (plusieurs interfaces peuvent être reliées à un instrument et une interface peut jouer plusieurs instruments)
3. utiliser les bus audio, video et gestuels pour faire interagir les instruments entre eux
4. connaître la librairie d'instruments audiovisuels proposés
5. utiliser leurs propres ressources numériques: sons, images, vidéos, volumes 3D, fichier MIDI
6. savoir déclarer de nouvelles interfaces gestuelles (USB, MIDI, OSC)
7. maîtriser les entrées et sorties audiovisuels de la plateforme (caméra, microphone, spatialisation audio, 3D relief)
8. savoir utiliser le système de mémoires dynamiques aux différents niveaux du logiciel

Cette plateforme puissante et souple est d'ailleurs utilisée dans la plupart des créations de PUCE MUSE. Elle ne nécessite pas de connaissances de programmation informatique.

Figure 1. Le logiciel Méta-Mallette 4.0

2.2. Le SDK

La MM4 utilise principalement le logiciel Max/MSP de cycling74. Elle est constituée de trois parties :

1. la plateforme avec la gestion des entrées sorties audiovisuelles, des interfaces gestuelles, le système de mémorisation et la circulation de données
2. les instruments virtuels audiovisuels qui vont utiliser différentes techniques pour produire du son et de l'image
3. les extensions qui ajoutent des fonctionnalités spécifiques à chaque projet artistique (intégration d'interfaces gestuelles spécifiques, égalisation des sorties audio ...)

Le SDK propose aux développeurs connaissant Max/MSP de développer leurs propres instruments et extensions. Il offre une série d'abstractions pour déclarer des lecteurs sons, des lecteurs images, des variables gestuelles... Son objectif est de simplifier au maximum tout ce qui concerne la connexion avec la plateforme pour laisser le développeur se concentrer sur l'algorithme audiovisuel spécifique à l'instrument. Un instrument simple peut être développé en quelques minutes.

2.3. La Mini-Mallette

La Méta-Mallette est très souvent utilisée collectivement : classe de formation musicale, de musique au collège, à l'école primaire avec des musiciens intervenants... Avec la naissance de projets artistiques plus ambitieux, des élèves ont demandé à pouvoir s'entraîner individuellement. La Mini-Mallette est une version plus simple acceptant au maximum 3 instruments audiovisuels simultanés. Elle nécessite donc beaucoup moins de puissance informatique pour fonctionner.

2.4. Le Player

La Méta-Mallette est puissante mais donc complexe pour un premier usage. Le player propose une interface simplifiée destinée uniquement à jouer des projets artistiques sans pouvoir les éditer. Cette demande est largement majoritaire chez les utilisateurs. En effet, concevoir un projet demande du temps et des compétences dans de nombreux domaines : musique, électroacoustique, image, vidéo, 3D... Le player propose donc de jouer des projets créés avec la Méta-Mallette et hébergés sur le site de la Méta-Librairie (voir paragraphe 5).

3. PROPOSER UNE COLLECTION D'INSTRUMENTS VIRTUELS SONORES ET VISUELS

3.1. La collection PUCE MUSE

Avec la plateforme Méta-Mallette, PUCE MUSE propose une collection d'instruments audiovisuels de base. Ces instruments peuvent fonctionner avec toutes les interfaces gestuelles reconnues dans la MM4 (USB, MIDI, OSC). Cependant seul un mapping par défaut est proposé pour joystick, gamepad, et souris. Tous les instruments fonctionnent maintenant en 3D relief (voir section 4). Un code couleur permet immédiatement de savoir à quelle famille chaque instrument appartient.

— 3 instruments jouant des sons échantillonnés
- *MM.Accel* est un instrument virtuel qui se base sur l'énergie du geste pour contrôler une synthèse par lecture d'échantillons.
- *MM.Scratch* est un instrument virtuel simulant le scratch sur platine vinyle.
- *MM.Groove* est un instrument virtuel qui permet de travailler en rythme tout en transposant.

— 1 instrument de synthèse « pure »
- *MM.FMot* se base sur la synthèse par modulation de fréquences (FM).

— 1 instrument dérouleur de fichier MIDI
- *MM.Roll* permet de faire défiler des fichiers MIDI en temps réel.

— 2 instruments de transformation (sons et images)
- *MM.Reve* est un instrument virtuel qui modifie son et image. Il simule l'effet de réverbération.
- *MM.Teeth* est un instrument virtuel qui modifie son et image. Il module un retard temporel du son et une réinjection visuelle.

— 1 instrument pour jouer des plugins audio
- *MM.VST* permet de contrôler avec les interfaces gestuelles des programmes au format VST.

— 1 instrument visuel pour jouer des vidéos
- *MM.Vignette* permet de travailler de la vidéo en temps réel.

3.2. Les instruments de synthèse vocale du LIMSI

Le LIMSI développe depuis plusieurs années des instruments vocaux, c'est-à-dire des systèmes de synthèse vocale temps-réel à contrôle gestuel [6]. La Méta-Mallette offre une plateforme d'intégration, de développement et de diffusion particulièrement intéressante pour ce type d'instrument, en particulier dans la dimension du jeu collectif, ou « chorale numérique ». Trois types d'instrument sont intégrés, ou en cours d'intégration, dans la Méta-Mallette :

1. Un synthétiseur de voyelles chantées, (avec contrôle des voyelles et de la source) utilisé en soliste ou en chœur (LIMSI.CantorDigitalis)
2. Un synthétiseur de (certaines) consonnes chantées, avec contrôle de l'articulation, utilisable en soliste ou en chœur (LIMSI.Digitartic).
3. Un système d'analyse-synthèse-modification temps-réel de l'intonation (LIMSI.Calliphony).

3.2.1. LIMSI.CantorDigitalis

LIMSI.CantorDigitalis est un instrument de synthèse vocale de type source-filtre utilisant la synthèse par formants, et comme source le modèle d'onde de débit glottique CALM [7,9].

Les quatre types classiques de voix (basse, ténor, alto et soprano) sont disponibles en ajustant pour chaque type : la tessiture ; les valeurs des filtres formantiques des voyelles ; la tension et le souffle de la source glottique.
Des perturbations de la source glottique sont ajoutées à différentes échelles temporelles : Jitter et Shimmer à l'échelle de la période fondamentale ; perturbation sinusoïdale décroissante sur la fréquence fondamentale et l'amplitude de vibration de la source, à l'échelle de la seconde pour reproduire l'effet de la pulsation cardiaque [21] ; extinction de la voix après une durée correspondant à l'utilisation du volume pulmonaire, grossièrement modélisé par une fonction de la force vocale au cours du temps.
Deux types d'interactions source-filtre sont introduits. D'une part, les amplitudes des filtres formantiques sont automatiquement ajustés en temps réel, afin de réduire les résonances excessives quand la fréquence fondamentale ou une de ses premières harmoniques coïncident avec les fréquences des filtres formantiques. D'autre part, les fréquences des filtres formantiques sont rendues dépendantes de la fréquence fondamentale, pour simuler la modification de la forme de la cavité buccale avec la fréquence fondamentale et la force de la voix [19].

LIMSI.CantorDigitalis [10] est configuré pour le contrôle de la production de voyelles chantées artificielles. A l'aide d'une tablette graphique, l'utilisateur contrôle la fréquence fondamentale le long de l'axe X et la force vocale avec la pression du stylet sur la tablette (fig. 2). Un mode permet de produire du chant diphonique en ajoutant une deuxième tablette graphique permettant de contrôler dans un plan 2D les fréquences des deux premiers formants ainsi que leur amplitude avec la pression de ce stylet.

Figure 2. La tablette graphique munie d'un clavier dessiné

Une étude en cours d'analyse sur la précision de la tablette graphique comme contrôle de l'intonation de la voix chantée est encourageante : elle tend à montrer qu'on parvient à produire des intervalles aussi, voire plus, justes avec la tablette qu'avec sa propre voix. Ces premiers résultats confirment une étude préliminaire [18] qui montre que la justesse nécessaire à un jeu musical de type chorale peut être atteinte à l'aide d'une tablette graphique.

3.2.2. LIMSI.Digitartic

LIMSI.Digitartic est une extension du synthétiseur précédent, conçu pour le contrôle de l'articulation de certaines consonnes et donc de syllabes voyelle-consones, consonnes-voyelles, ou composées. Il possède un module supplémentaire, par rapport à LIMSI.CantorDigitalis, sous la forme de règles sur les trajectoires des formants lors des transitions de phonèmes, l'aspiration et le bruit des occlusions et frictions. La première version de cet instrument permet de contrôler continument le lieu d'articulation et l'instant articulatoire de certaines consonnes liquides, semi-voyelles et occlusives à l'aide d'une tablette graphique. Le but est d'en faire un instrument de synthèse vocale capable de contrôler de la musique de type *scat*.

Ces deux instruments, LIMSI.CantorDigitalis et LIMSI.Digitartic, vocaux, peuvent être joués en solo ou en chœur. Une chorale numérique, le *Chorus Digitalis* [10,18], se réunit actuellement toutes les semaines (fig.3). Le but est ainsi double : les instruments sont d'une qualité suffisante pour produire de la musique, et d'autre part, le jeu collectif est un outil pour l'étude scientifique de tels contrôles gestuels de la synthèse vocale.

Figure 3. Le Chorus Digitalis

3.2.3. LIMSI.Calliphony

LIMSI.Calliphony, n'est pas un synthétiseur, mais un système d'analyse-modification-synthèse de l'intonation et du rythme de la prononciation. Il permet ainsi de modifier en temps-réel la prosodie d'une phrase en contrôlant la fréquence fondamentale de la voix et l'instant de lecture d'un fichier audio de voix pré-enregistrée.
On utilise une interface délivrant des coordonnées continues suivant deux dimensions et possédant une résolution spatiale suffisante, comme une tablette graphique. L'axe X est relié à l'instant de lecture de la phrase et l'axe Y à l'intonation. La précision obtenue pour contrôler l'intonation [8] à l'aide d'une tablette graphique et d'un tel système est supérieure ou égale à celle obtenue par la voix. La capacité à contrôler une intonation précise avec la tablette graphique provient de notre habileté à manier un stylo, issue de l'apprentissage précoce et assidu de l'écriture. L'instrument est basé sur une implémentation temps réel de l'algorithme PSOLA [15] [17] qui permet la modification de la fréquence fondamentale et l'étirement/contraction temporel d'un fichier de voix.
Cet instrument, conçu au départ pour la recherche en prosodie de la parole, est aussi susceptible d'utilisations musicales, par la modulation rythmique et mélodique de textes ou de voix enregistrés.

3.3. Les instruments par Modèles Intermédiaires Dynamiques (MID) de l'équipe Lutheries acoustique musique (LAM)

Dans le cadre du projet OrJo, le LAM développe de nouveaux instruments pour la Méta-Mallette sur la base des recherches menées par le LAM autour de la notion d'instrumentalité des outils de production musicale [3] [11], et prenant en compte les réflexions théoriques conduites depuis plusieurs années dans le champ des Interfaces Homme-Machine [1][4], en particulier dans le domaine musical [5] [20] [22].
Ces instruments sont basés sur le concept de MID (Modèle Intermédiaire Dynamique), concept présenté lors des JIM 2011 à Saint-Étienne [13] et développé à Padova à l'occasion de SMC 2011 [14]. Ces modèles s'appuient sur une architecture logicielle modulaire pour dépasser le traditionnel *mapping* [16] en ajoutant des éléments logiciels dotés de comportements dynamiques (modèles physiques, topologiques, statistiques, etc.) dans la chaîne de contrôle de la synthèse.

Les objectifs visés par l'utilisation des MID sont de différentes natures et conduisent à élaborer un cahier des charges en décrivant les aspects fonctionnels et structurels. Sans être exhaustifs, nous évoquerons cependant quelques-unes de leurs caractéristiques essentielles :

- « augmenter » le geste en générant, à partir des données issues des capteurs, des flux de paramètres de contrôles (travail à des fréquences intermédiaires entre contrôle gestuel et fréquence du signal audio)
- s'intégrer dans une architecture logicielle modulaire respectant le caractère empirique de toute forme de lutherie (pouvoir tester différents types de synthèse à partir d'un même MID ou vice-versa)
- offrir une panoplie de modules correspondant à des besoins variés (échelles temporelles et logicielles en particulier)
- offrir plusieurs niveaux de monitoring et si nécessaire, gérer avec les mêmes algorithmes différentes modalités sensorielles.

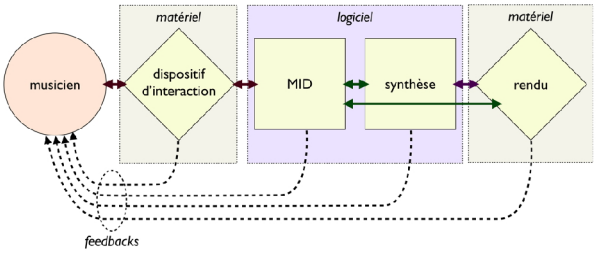


Figure 4. Schéma fonctionnel illustrant la place d'un modèle intermédiaire dans la chaîne allant du musicien au dispositif de restitution sonore.

4. AMÉLIORER LA REPRÉSENTATION GRAPHIQUE DES INSTRUMENTS VIRTUELS

La représentation visuelle des instruments se fait dans un espace 3D (opengl). Chaque instrument peut générer sa représentation dans cet espace virtuel en utilisant les fonctionnalités habituelles telles que sa position, la position de la caméra virtuelle, la lumière...

OrJo ajoute la possibilité de travailler en relief en utilisant des lunettes anaglyphiques. Cette technique permet de montrer des images 3D sur tous supports et les lunettes utilisées sont très économiques.
Voici quatre champs d'interactions entre son et images reliefs qui semblent particulièrement fertiles.

4.1. Le plaisir de la contemplation

Où le spectateur promène son regard librement dans un paysage virtuel, où il s'enfonce loin au delà de l'écran. Cette usage est magnifiquement illustré par le dernier film de Wim Wenders sur Pina Bausch. L'interaction avec le son spatialisé et l'écoute intelligente est évidente.

4.2. L'attirance pour le vertige

Ne plus savoir les limites du cadre, décoller de l'écran des morceaux d'instruments jusqu'à tendre la main pour les attraper. Le vertige visuel peut amplifier l'écoute. En effet la perception demande alors à l'audition de préciser

l'incertitude visuelle. L'écoute acousmatique fonctionne aussi sur ce principe.

4.3. Du réel au virtuel

L'interaction avec les caractéristiques physiques de l'écran, qu'il soit vidéo-projeté ou simple écran d'ordinateur, permet la création d'instruments se situant entre réel et virtuel. Il s'agit de prendre en compte le volume, les contours, teintes, matières de l'écran comme autant de guides pour concevoir un instrument. L'exemple le plus simple est celui de particules virtuelles qui rebondissent sur les bords de l'écran réel en générant des sons au moment où elles rebondissent. Plus les propriétés physiques de l'écran sont non neutre plus l'instrument virtuel peut interagir avec lui et offrir une singularité acoustique. Le relief visuel amplifie encore cette fluidité entre réel et virtuel.

4.4. Amplifier l'écoute

En associant sons et formes visuelles en mouvement, de nouvelles fonctionnalités apparaissent :

- Amplifier le geste : il est possible d'agrandir le geste musical ou le corps de l'instrument.
- Pré-voir et post-voir : la partition permet d'anticiper sur ce qui va arriver comme elle peut porter la trace de ce qui vient de se jouer. La musique *poly-phonique* devient alors *poly-graphique*. Ces représentations deviennent des guides qui structurent la mémoire du spectateur. Elles associent un vocabulaire graphique (souvent beaucoup plus étendu que le vocabulaire sonore) pour commenter et mémoriser la musique. Elles aident l'auditeur à suivre l'architecture musicale. Sur l'ensemble de ces fonctionnalités le relief agrandit le champs des possibles en même temps qu'il agrandit l'espace de la représentation.

5. PRATIQUER, ÉCHANGER, ET CONSERVER UN RÉPERTOIRE SUR LA MÉTA-LIBRAIRIE

La Méta-Librairie (http://www.meta-librairie.com) est un site d'échange fonctionnant sur le modèle de l'appstore. Les projets artistiques, instruments virtuels et extensions peuvent être « uploadés et downloadés » par les utilisateurs de la plateforme. Les modules téléchargés peuvent être gratuits ou payant, programmables – à la manière d'une œuvre ouverte, ou juste exécutables – telle une partition à jouer.

Figure 5. Page d'accueil de la Méta-Librairie.

Pour noter et documenter ces projets, plusieurs formats et codes ont été progressivement formalisés. L'intérêt de cette normalisation est de simplifier l'échange de projets.

5.1. La notation tablature

Cette notation symbolise les mouvements à réaliser avec les interfaces gestuelles choisies. Elle précise les parties mobiles de l'instrument dans le temps et la latitude des joueurs. Souvent une notation graphique se superpose à cette notation « tablature ». Elle donne une indication du résultat sonore. Une planche de symboles et codes utilisés est disponible sur la Méta-Librairie.

5.2. La vidéo

La vidéo est probablement la meilleure notation / transmission de projet artistique. Sur la vidéo sont représentés :

- le résultat audio et visuel de chaque instrument,
- en incrustation les mouvements du joueur avec son interface gestuelle,
- une incrustation de la partition avec le déroulement d'un pointeur sur la partition.

Ce système permet de travailler sa partition en superposant tout ou partie de la vidéo avec le jeu en direct de l'instrument.
Enfin une réalisation complète de l'œuvre filmée est ajoutée quand l'œuvre a été filmée en concert.

5.3. Une direction inspirée du sound painting

Le *sound painting* est un langage de composition en temps réel créé par Walter Thompson dans les années 1980. Plusieurs gestes ont été créés pour diriger des orchestres de joysticks et permettre de jouer rapidement en orchestre. Une vidéothèque répertorie les gestes spécifiques utilisés pour la Méta-Mallette.

Progressivement le fonds PUCE MUSE sera mis en ligne sur la Méta-Librairie. Il comporte deux catalogues.
Le premier recense des œuvres de 2003/2010. Il est constitué d'une vingtaine d'œuvres pour manettes de jeux vidéos type joystick, gamepad... Ce sont, pour la plupart, des pièces à jouer en ensemble.

Le second recense des œuvres de 1989 à 2004. Il est constitué d'une trentaine d'œuvres pour des interfaces gestuelles professionnelles comme le Méta-Instrument. Ce sont, pour la plupart, des œuvres solistes.

6. CONCLUSION

Au delà du projet OrJo, nous souhaitons que la Méta-Librairie devienne un site de mutualisation des projets artistiques, scientifiques et pédagogiques pour mieux comprendre ce champs artistique que nous appelons la Musique Vivante Virtuelle Visuelle (M3V). Il sera donc possible de poster des articles, de mettre des liens vers des projets ou des concerts qui questionnent ce champs artistique. En effet, il est indispensable de multiplier et thésauriser les projets associant multi-modalité, pratique collective et répertoire artistique pour les nouvelles technologies afin de comprendre et partager ce champs artistique. La Méta-Librairie propose donc d'être un centre de documentation, un carrefour d'idées, de réflexions, de questionnements, d'échanges autour de ces thématiques. D'ailleurs, le mot anglais *library* ne veut il pas dire *bibliothèque* ?

7. RÉFÉRENCES

[1] Beaudouin-Lafon, M., « Moins d'interface pour plus d'interaction », Interfaces Homme-Machine et Création Musicale, H. Vinet et F. Delalande (éds), Hermès, 123-141, 1999.

[2] Cadoz, C., « Musique, geste, technologie », Les nouveaux gestes de la musique, H. Genevois et R. de Vivo (éds), Parenthèses, 47-92, 1999.

[3] Cance, C., Genevois, H., Dubois, D., « What is instrumentality in new digital musical devices? A contribution from cognitive linguistics and psychology », Proc. of CIM09, à paraître « La musique et ses instruments », Delatour, 2012.

[4] Castagne, N., Cadoz, C., « 10 criteria for evaluating physical modelling schemes ». in proc. DAFX'03.

[5] Couturier, J-M., « Utilisation avancée d'interfaces graphiques dans le contrôle gestuel de processus sonores », thèse, Univ. de la Méditerranée, Marseille, 2004.

[6] d'Alessandro, C. D'Alessandro, N. Le Beux, S. Simko, J. Çetin, F. Pirker, H. "The Speech Conductor: Gestural Control of Speech Synthesis" Proc. eNTERFACE 2005, Mons, 52-61, 2006.

[7] D'Alessandro, N. d'Alessandro, C. Le Beux, S. Doval, B. "Real-time CALM Synthesizer New Approaches in Hands-Controlled Voice Synthesis", Proc. NIME'06, 266-271, Paris.

[8] d'Alessandro, C., Rilliard, A., Le Beux, S. « Chironomic stylization of intonation » J. Acoust. Soc. Am., 129(3), 1594-1604, 2011.

[9] Doval, B., d'Alessandro, C., Henrich, N. « The voice source as a causal / anticausal linear filter ». Proc. of Voqual'03, Geneva, 2003.

[10] Feugère, L., Le Beux, S., d'Alessandro, C. « Chorus digitalis : polyphonic gestural singing », Proc. 1st Int. Workshop on Performative Speech and Singing Synthesis, Vancouver, 2011.

[11] Genevois, H., « Geste et pensée musicale : de l'outil à l'instrument », Les nouveaux gestes de la musique, H. Genevois et R. de Vivo (éds), Parenthèses, 35-45, 1999.

[12] Goudard, V., De Laubier, S. « PUCE MUSE – La Méta-Mallette », Proc. JIM 2008.

[13] Goudard, V., Genevois H., Ghomi E. « L'utilisation de modèles intermédiaires dynamiques pour la synthèse audio-graphique » Proc. JIM 2011, Saint-Etienne.

[14] Goudard, V., Genevois, H., Ghomi, E., Doval, B., « Dynamic Intermediate Models for Audiographic Synthesis », SMC 2011, Padova, Italie, 2011

[15] Hamon, C., Moulines, E., Charpentier, F. « A diphone synthesis system based on time-domain prosodic modifications of speech », Proc. IEEE-ICASSP'89, 238-241.

[16] Hunt, A., Wanderley, M. M., Paradis, M., « The importance of parameter mapping in electronic instrument design ». Proc. NIME'02.

[17] Le Beux, S. Doval B. and d'Alessandro, C (2010). "Issues and solutions related to real-time TD-PSOLA implementation." 128th Conv. of the AES London, 2010.

[18] Le Beux, S. Feugère, L. D'alessandro, C. « Chorus digitalis : experiment in chironomic choir singing », Proc. Interspeech 2011.

[19] Liénard, J.-S., Di Benedetto, M.-G. « Effect of vocal effort on spectral properties of vowels », J. Acoust. Soc. Am. 106 (1), 411-422, 1999.

[20] Momeni, A., Henry, C., « Dynamic Independent Mapping Layers for Concurrent Control of Audio and Video Synthesis », Comp. Music J., . 30(1), 49-66, 2006.

[21] Orlikoff, F. R. « Heartbeat-related Fundamental Frequency And Amplitude Variation In Healthy Young And Elderly MaleVoices », J. of Voice vol. 4, No 4, 322-328,1990.

[22] Stowell, D., Plumbley, M.D. & Bryan-Kinns, N., « Discourse analysis evaluation method for expressive musical interfaces », Proc. NIME'08, Genova.

KINECTIC WAVES AT ART ZOYD STUDIOS

Carl Faia
Art Zoyd
cf@carlfaia.com

Nadia Ratsimandresy
Art Zoyd
nratsimandresy@hotmail.com

Abstract

This paper describes the work in progress of the project undertaken at Art Zoyd Studios to create a new work for Nadia Ratsimandresy and the ondes Martenot (*ondes*)with composer and developer Carl Faia. A brief historical background of the instrument will be given, details of the Max (cycling74) patch will be provided and the directions of development already explored as well as those to possibly be explored. While part of the project is focused on developing the repertoire for the *ondes*, we will present current research and development towards the creation of an augmented instrument allowing the player to expand the playing modes of the unique device and develop an advanced virtuosity by connecting the instrument to the computer, with Max, and using a tracking system combining audio analysis, pitch and amplitude tracking with Max external sigmund~, and visual gestures tracked with the Microsoft Kinect® video camera.

1. INTRODUCTION

Invented in 1928 by Maurice Martenot, the *ondes Martenot* (*ondes*) is one of the first examples of a sound synthesizer. Its unique status - electronic instrument with a strong acoustic element (control of dynamics, phrasing, tuning, timbre, sound diffusion and vibrato) has aroused the interest of many composers. The repertoire has been enriched by more than 2000 musical pieces. From Olivier Messiaen to Tristan Murail, through Darius Milhaud, André Jolivet, Jacques Brel, *Radiohead*, a very large stylistic repertoire has been built.

The instrument is a strange combination and precociously advanced application of traditional playing techniques (with the integration of vibrato through a loosely fitted keyboard) and avant-garde electronic effects (noise generator, random frequency distortions or the unique ring-on-a-string highly accurate pitch controller). Then there is the curious and rich sound diffusion available through the player's ability to change the output timbre through speaker choice and combinations — on the fly! Figure 1 shows the control panel. In fact, there are so many playing techniques and variations, a composer might feel daunted by the sheer number of possibilities the instrument offers and thus prefer to approach the instrument as a novelty or as a "simple" keyboard instrument.

The growing importance of renewing the repertoire, as well as expanding the place of the *ondes* in the contemporary music scene, is, in part, the impetus behind Nadia Ratsimandresy's demand to *Art Zoyd* [1] to have a new look at the instrument and work on developing the repertoire of an instrument on the comeback.

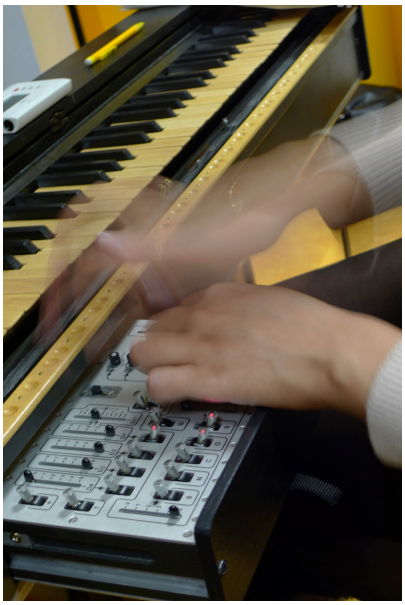


Figure 1. Photo of keyboard and control panel of the *ondes*

While the various parts of the *ondes* are being investigated and experimented with for revision or perfection, the most notable for us is the new speaker, as seen in Figure 2, as both its shape and the various audio connections are unique and convenient for the developments we are currently exploring. Inputs and outputs have been multiplied and we can easily connect and route various audio leads to our interface inputs.

Figure 3 shows the so-called palm speaker – with strings tuned to resonate with the transformer vibrating the wood panels – as yet untouched by Nadia's team, and the "gong" speaker stripped and in the development process. While this is in itself an interesting process and worth further description, this paper will deal with the computer music side of the project. More may be found on further developments in future instalments on the Art Zoyd website.

Figure 2. Photo of new speaker developed for the *ondes* for Nadia Ratsimandresey

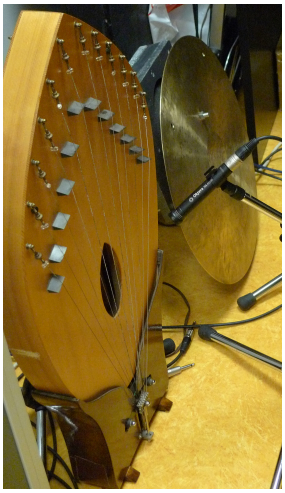


Figure 3. Photo of palm speaker and the new gong/reverb diffusion system being created for the *ondes*.

2. THE PATCH

Figure 4 shows a screenshot of the Max patch currently in development for the *ondes* project. The patch itself is often called, The Patch. It is a constantly evolving collection of modules being developed by Carl Faia for both pedagogical and creative ends. The patch is structured around a matrix, or virtual patchbay, used to interconnect everything within the patch. The idea being that anything might be connected to anything else with a click of the mouse, and then saved in a preset for later recall. Of particular interest in this patch is the controller interface being developed, as seen in Figure 5, allowing for any kind of external controller to be input, scaled, smoothed and then sent through remote messages to any module in the patch. Figure 6 shows an example of this. While a simple reverb effect, the parameters are dynamically manipulated through assigned controller numbers (i.e. 1-Ic3 for interface 1, controller 3) thus, potentially, creating a dynamic complex instrument.

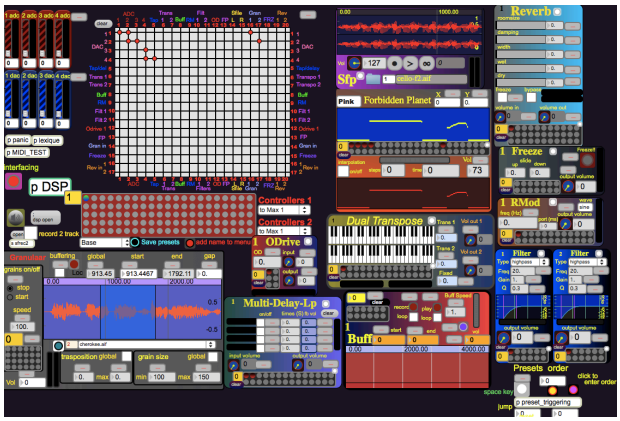


Figure 4. Max patch being developed for the work

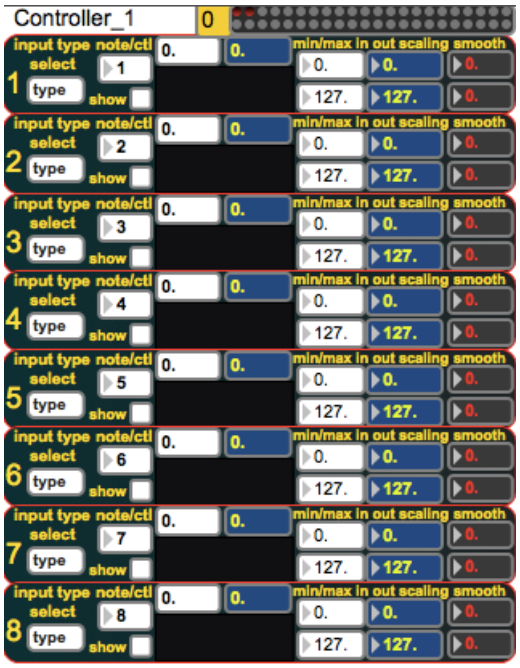


Figure 5. Detail of controller interface for the patch

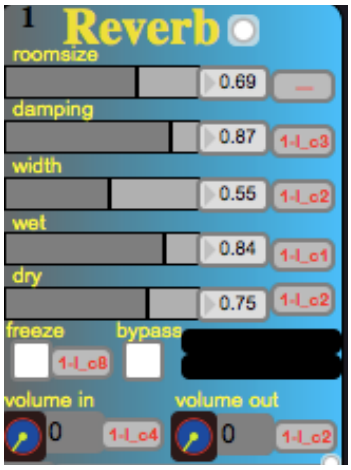


Figure 6. Detail of one effect with controllers assigned in the menus.

3. THE KINECT AT ART ZOYD

Using the Kinect® at Art Zoyd, Figures 7 and 8, started out with composer André Serre-Milan on another project still in progress [2]. Carl Faia started using the video and, more importantly, the skeletal data available from the relatively new, and complicated, process described in many places but nicely outlined in Jon Bellona's white paper on the subject [3]. As usual with any new forms of sensors being hacked or bastardized into processes for which they were not originally meant, there is a period of adaption and the first steps are often extremely basic or clichéd (we saw this with the use of the Wii, but also the joysticks and other game controllers). What became immediately obvious with the Kinect® device was the unprecedented preciseness of the information being delivered (one may easily create a very accurate "theremin").

While all this would be great and fine, there was the problem that getting the system to work reliably and easily was, until recently, not certain. Using this kind of technology might never, indeed, be certain; however there has been a new development that brings it closer to being more certain than before (especially as before, one needed to be running loads of programs through the Terminal on Max OS X). This has come about after the first 2 periods of residence in Art Zoyd Studios so has yet to be explored, but the ease of using the Kinect® in concert is now more probable by the arrival of *Ni mate* [3]. Still in beta, this program, developed by *Delicode*, seems to have solved the major obstacles to using the Kinect® in performance, most notably the ability to automatically find the user and to communicate easily and quickly with Max.

The idea is to have the performer controlling various parameters of the Max patch through the data provided by the Kinect® through *Ni mate* into the controller interface of the patch and then on to the parameters assigned through the menus. All of this data is, at any point along the path, scalable and malleable.

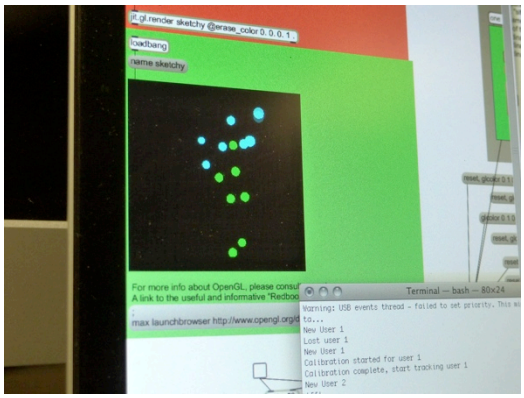


Figure 7. One of the first examples of the Kinect® controlled graphics developed at Art Zoyd Studios.

Figure 8. The Kinect "in use".

4. CONNECTING IT ALL TOGETHER

Technically a flow diagram would have the performer at the top influencing all other parameters. In this project, she will be controlling the sound with her playing of the instrument to produce the raw output, but she will also be controlling the sound through the movement of her body and, in some of our preliminary studies, with the pitch and amplitude of the instrument's analysed raw outputs.

4.1. Analysis with sigmund~

Using the pitch tracker to follow the sinusoidal output of the *ondes* is an obvious use of the instrument's fundamental sound. It is sinusoidal or saw form by choice of the performer. Both are relatively easy to analyse with readily available third party Max objects such as sigmund~ [4]. The pitch and amplitude data can then be used in the same way as any external controller and mapped in through the built in controller of the patch.

4.2. Kinect and the patch

The project, as it is now being formalized, will have a multiform augmented sound-producing element, the *ondes,* capable of creating sound through the various methods related above, as well as through the treatments and/or synthesis of the patch. The use of the analysis of the sound made/played adds a layer of control through analysis to the musical environment, as does the use of motion capture to analyse/recognize and "interpret" the performers movement.

5. CREATING A ONE WOMAN SHOW

We are looking at creating a full length concert/spectacle including a new work written for Nadia and this augmented instrument, a classic arranged for her and this new environment and a work of improvisation created in a collaboration between Ratsimendresy and Faia.

5.1. A new work for the repertoire

The new work will be created from the close collaboration the two musicians have had in the studio. The work will be presented in score form and be of around 15 minutes in duration. The various bits of technology as described above will be used throughout in a creative fashion. As of this writing, live video is planned as an option. Our intention is to create a new work that will be both general for the repertoire of the *ondes*, and playable by other ondistes without too much difficulty in the technological sense, but also particular enough that Nadia's unique approach to the instrument will be preferred and highlighted.

5.2. An old work revisited

As an important part of the project, the classic work by Karlheinz Stockhausen, *Solo for a melodic instrument* (1965-66), will be arranged for the *ondes* by Carl Faia. This is a work he knows well and has already developed with other performers [5] [6]. The patch will have the *Solo* software development integrated into the environment so that the entire system may be included in the finalized version. While this was not an integral part of the project, it has become a fitting inclusion in the collaboration. The openness composed into Solo leaves much to interpretation and the possible effects indicated in the directions are without any real constraints… as if the composer was leaving it all open for these current developments.

6. CONCLUSION

Creating something new for *ondes* is what started this project. We wanted to go further and not only add something to the repertoire but add something new to the instrument itself. While this may or may not be a permanent addition, it does integrate nicely into the overall wish of the performer to reinvent the instrument she is playing and of the composer/developer to augment, through external and internal means — Kinect® and analysis with sigmund~ — the typical solo instrument with electronics piece. In the process, we are both learning how the other works because we are often stuck in the studio together learning how each other's specialized instrument work (the *ondes* and "the patch") and this creates a multi-layered richness to the experience and, in turn, the work tat comes out of this experience. The addition of Stockhausen's *Solo* rounds out the experience and will, we both hope, help instil a creative link back half a century to a certain raw period in the development of electronic music and even further back, nearly a century, to the instrument dreamed up and created by Marice Martinot.

Finally, all this work towards new repertoire, improvisational development, and augmented instruments lies firmly within the stated and implied charter of Art Zoyd Studios since the opening of the studio side of the group in 1997: the development of the visual element in the spectacle, making new instruments be they virtual or physical (*luthier*), interest and projects based around interfacing the body and/or the instrument with the computer, long term collaborations and a curiosity and knack to bring the old back from limbo and breathe new life into old forms (silent films are an example, but so is the violin-Stroh or the Theremin and the *ondes Martinot* used in concert as if they were just any other instrument) [7].

It is becoming more and more apparent that the Kinect® might very well be the connecting element we've been looking for since many years. With the latest developments in the software integration and the relatively low cost and ease of acquiring the technology, we should be quickly developing new forms of performance practice. The project for *ondes* has been a pivotal experience both for us and, by implication, for other current and future projects in Art Zoyd Studios.

7. REFERENCES

[1] Art Zoyd, Nadia Ratsimendresy & Carl Faia – Residence in October 2011 http://www.artzoyd.net/2011/10/nadia-ratsimendresy-carl-faia-en-residence-en-octobre-2011/?lang=en.

[2] Kinect …ing, video, YouTube, 19 February 2011, viewed 10 March 2012, http://www.youtube.com/watch?v=IAs4VKjvMEo.

[3] Ni Mate, Delicode, http://www.ni-mate.com.

[4] sigmund~, sinusoidal analysis and pitch tracking by Miller Puckette, MSP port by Miller Puckette, Cort Lippe, Ted Apel http://crca.ucsd.edu/~tapel/software.html.

[5] Sluchin, Benny, "A Computer-Assisted Version of Stockhausen's Solo for a Melody Instrument with Feedback", Computer Music Journal, Volume 24, no. 2 (2000): pp 39–46.

[6] Expériences de Vol #8, 2011, audio CD, Orkhêstra International.

[7] The Histroy of Art Zoyd, viewed 10 March 2012, <http://www.artzoyd.net/histoire-art-zoyd>.

SEQUENZA

Denis Pousseur
denis.pousseur@gmail.com

Le nom « Sequenza » est un hommage à Luciano Berio

RÉSUMÉ

Sequenza [1] est un nouveau logiciel d'aide à la composition qui arrive à sa première version publique après une huitaine d'années de recherche et développement.
Sequenza se situe quelque part à la frontière entre deux grandes familles logicielles : les séquenceurs MIDI et les éditeurs de partition. Il possède aussi un important volet d'aide à la création, s'approchant ainsi de certains outils de recherche comme, par exemple, *Open Music* [2] développé à l'IRCAM.

L'objet de cet article est de présenter ce positionnement et la manière dont le polymorphisme qui en découle est mis en application dans quelques aspects concrets de l'application. L'article commence par des considérations de bas niveau touchant à certains aspects fondamentaux de la théorie musicale, puis évolue vers des questions touchant aux plus hauts niveaux de l'application, tout particulièrement à son architecture.

Figure 1. *Le logo de Sequenza*

1 . Sequenza est une application qui tourne exclusivement sur Mac OS X, version 10.5 (Léopard) ou ultérieur.

2 . Les recherches qui ont mené progressivement au développement de Sequenza ont commencé sur la plateforme *Open Music*. Ceci explique certaines convergences de conception même si, aujourd'hui, les philosophies des deux applications sont fort différentes, et de ce fait d'ailleurs assez complémentaires. Ceci explique aussi pourquoi les couches fondamentales de Sequenza sont développées en Common Lisp comme l'est *Open Music*. À noter toutefois que la couche de présentation (GUI) de Sequenza est entièrement basée sur la bibliothèque Cocoa de Mac OS X et fait donc usage de l'Objective-C. Ceci est rendu possible par le bridge entre Lisp et Objective-C développé à Cambridge par la firme Lispworks (2008) [1].

1. INTRODUCTION

Un des objectifs de Sequenza est de permettre au compositeur de travailler sur les rendus sonore et graphique d'une partition dans un même environnement tout en lui offrant de nombreux outils d'assistance à la composition.

À la différence d'*Open Music*, Sequenza n'est toutefois pas un « métalangage » de programmation. En se positionnant au carrefour de plusieurs familles logicielles, Sequenza ne peut en effet pas développer de manière exhaustive chacune des matières qu'il traite. Il jette plutôt entre elles des passerelles qui ne sont possibles que dans un environnement intégré. Ces passerelles structurent l'application qui cherche ainsi à s'approcher au plus près du « geste » compositionnel.

L'attention portée par Sequenza à la modélisation sonore accompagne une mutation qui, peu à peu, confère à l'ordinateur le rôle que les compositeurs donnaient autrefois au piano. Tout éditeur de partition propose aujourd'hui une forme de modélisation sonore, mais il le fait en s'appuyant sur des automatismes basés sur l'iconographie sur lesquels l'utilisateur a peu de contrôle. Cette approche est attractive (la partition se joue « toute seule ») mais le résultat en est généralement peu convaincant.
Sans dédaigner cette voie, Sequenza développe d'autres stratégies. L'accès aux données sonores y est tout aussi complet qu'il l'est dans un séquenceur MIDI et de nombreux mécanismes d'interaction entre l'écrit et le sonore sont proposés. Cet aspect est évoqué à la section 4.

Sequenza peut aussi se voir comme un « générateur et processeur de musique écrite ». On peut l'utiliser à la manière d'un créateur de musique électronique : travailler la matière, la générer, la traiter, l'organiser. On peut ensuite se poser des questions plus spécifiquement liées à l'écriture et à l'interprétation. Cette façon de voir un cycle de travail dans Sequenza explique l'attention portée à l'indépendance du rendu sonore et du rendu graphique, elle explique aussi l'attention portée à l'architecture et au capacités d'organisation des matériaux évoqués à la section 5.

Enfin, Sequenza aborde les questions de théorie musicale avec la volonté d'y intégrer les apports nombreux dont le XXe siècle l'a enrichie, ce qui est particulièrement illustré par les sections 2 et 3.

2. GESTION DES HAUTEURS

Sequenza s'efforce de rechercher des éléments de convergence entre les différentes pratiques compositionnelles. Pensé pour aider le compositeur d'aujourd'hui, Sequenza ne délaisse pas pour autant une gestion classique de la tonalité.
Dans ce même esprit, Sequenza tente une approche de la microtonalité qui soit compatible avec son versant tonal.
À ces approches tonale et microtonale, Sequenza ajoute encore les approches spectrale et modale, le tout formant un ensemble qui se veut le plus transparent possible pour l'utilisateur.

2.1. Hauteurs microtonales

2.1.1. Principe

Sequenza gère la microtonalité sous forme de deux modes s'ajoutant au mode chromatique compatible avec la norme MIDI (MIDI, 1982) [2] : les quarts de tons et les sixièmes de tons. Le choix de ces deux modes n'est pas arbitraire, il est lié aux choix fait pour la représentation graphique des micro-intervalles. Afin de créer une progression fluide entre sa dimension tonale et microtonale, Sequenza exprime les micro-intervalles sous forme de trois pas relatifs aux hauteurs chromatiques :

(-1, 0, 1) Hauteur baissée, normale ou haussée

Chaque type d'altération possède donc deux versions graphiques alternatives à la version normale, pour pouvoir s'adapter à ces degrés intermédiaires. La valeur haussée et la valeur baissée sont représentées par une petite flèche montante ou descendante au sommet ou au bas de l'altération (ou du bécarre pour les notes non altérées). Le double dièse et le double bémol n'ont respectivement qu'une version baissée et une version haussée.
Pour l'expression textuelle de la hauteur, les valeurs haussées et baissées seront symbolisées par un signe « + » ou « – » ajouté à la suite du nom de la note. Par exemple C♯3+ signifie un quart de ton ou un tiers de ton au-dessus de C♯3, ceci dépendant du mode microtonal choisi.

2.1.2. Mise en pratique

Si le mode microtonal choisi est « quarts de ton », la valeur haussée et la valeur baissée du degré chromatique supérieur sont simplement en « enharmonie » (c'est-à-dire que leur rendu sonore est identique).
Si le mode microtonal est « sixièmes de ton », la valeur haussée représente le premier tiers de demi-ton et la valeur baissée du degré chromatique supérieur représente le second tiers de demi-ton.
De cette manière, Sequenza est capable de mélanger la logique d'écriture tonale avec l'apport de la microtonalité. Par exemple, un quart de ton au-dessus de do peut se représenter au choix par :

si♯+, do+, do♯-, ré♭-, ou même ré♭♭+

2.1.3. Rendu sonore de la microtonalité

Les ingénieurs d'Apple ont prévu dans le protocole de communication avec le « synthétiseur DLS » de la bibliothèque « Core Audio » un format MIDI « étendu » (Apple Inc. 2001) [3] qui permet l'utilisation de valeurs fractionnelles en lieu et place des valeurs entières prévues par la norme MIDI. La hauteur reste codée comme un chiffre allant de 0 à 127 mais, grâce à la partie fractionnelle, on peut exprimer les valeurs microtonales en allant d'ailleurs bien au-delà de la définition offerte actuellement par Sequenza. Sequenza utilise cette fonctionnalité pour le rendu sonore de la microtonalité. Quand on travaille avec des instruments virtuels d'autres constructeurs, Sequenza représente les hauteurs microtonales, mais elles ne peuvent pas être entendues en raison des limitations du protocole MIDI.
L'ajout d'un système plus universel, par exemple inspiré de celui développé par OM (OM 6.5, Microintervals) [4], reste une solution possible pour le futur.
On peut aussi rêver d'un protocole MIDI « étendu » qui se généralise (HD-MIDI, 2008) [5], ou d'une reconnaissance plus répandue de l'extension à la norme prévue dès 1995 par la MMA (MIDI Tuning Messages, 1995) [6] afin de permettre la représentation d'échelles de hauteur non dérivées du système tempéré occidental.

2.1.4. Développements possibles

La limite de définition microtonale est fixée pour des raisons liées à l'expression graphique des altérations. Mais une version mieux définie du même système pourrait être envisagée en ajoutant les concepts de « demi haussé » et « demi baissé ». Ceci enrichirait l'offre des modes existants aux huitièmes et dixièmes de ton sans briser pour autant la compatibilité tonale.

2.2. Hauteurs tonales

La hauteur « tonale » (j'entends pas là, une hauteur qui - à la différence de la hauteur MIDI - inclut son expression diatonique et son altération) est indispensable à la représentation graphique de la partition. Pour stocker cette hauteur, Sequenza utilise un système de codage qui lui permet de la manipuler ensuite avec simplicité.
Dans Sequenza, ce que l'on appelle généralement « transposition tonale » (Letz, Fober, Orlarey, 1999) [7] est nommé « transposition modale » (un concept évoqué au point 2.4), alors que la transposition tonale recouvre un concept différent : c'est une « transposition chromatique » qui présente en plus l'avantage de préserver le dessin diatonique et la logique des altérations, c'est-à-dire la logique d'écriture. Concrètement, la hauteur de la note est codée, hors octave, dans une plage allant de 2 à 32 et représentant 3 cycles des quintes (incomplets) en même temps que 5 cycles diatoniques (incomplets). Fa est 14, do est 15, sol est 16… fa♯ est 21, fa♯♯ est 28, etc. (voir figure 2). Plus on monte plus on va vers des altérations « haussées », et

inversement. C'est un système assez proche de ceux évoqués par Rasch (2000) [8].

0	(F♭♭)	(C♭♭)	G♭♭	D♭♭	A♭♭	E♭♭	B♭♭
7	F♭	C♭	G♭	D♭	A♭	E♭	B♭
14	F	C	G	D	A	E	B
21	F♯	C♯	G♯	D♯	A♯	E♯	B♯
28	F♯♯	C♯♯	G♯♯	D♯♯	A♯♯	(E♯♯)	(B♯♯)

Figure 2. *Tableau de codage des altérations*

Dans l'espace décrit par la figure 2, les manipulations se font par de simples additions et soustractions d'index. Les notes entre parenthèses sont hors des limites acceptées. Hausser ou baisser une altération d'un degré revient à ajouter ou à soustraire 7 (déplacement vertical dans le tableau). Trouver l'enharmonie d'une note revient à ajouter ou à soustraire 12 ou 24 au code.
Tant que l'opération ne mène pas le code en dehors de la plage $[2, 32]$, elle est « possible » (au sens de la notation conventionnelle). Sinon, elle n'est possible qu'en remplaçant une hauteur par son enharmonie.
Par exemple, si une valeur devient 34 après manipulation (Si♯♯ – une valeur non représentable car $34 > 32$), il faut la ramener dans la plage valide en lui soustrayant 12. Elle devient donc 22 c'est-à-dire do♯, l'enharmonie la plus proche.
Du point de vue de l'utilisateur, la transposition est présentée de manière conventionnelle (ajout d'une seconde diminuée ou d'une quinte augmentée par exemple). Quand Sequenza réalise une « transposition tonale » explicitement et qu'une impossibilité est rencontrée, l'utilisateur a le choix entre abandonner la procédure ou accepter les enharmonies nécessaires au calcul.
Finalement, après traitement, on peut déduire des codes résultants les notes diatoniques (hors octave) et les altérations, c'est-à-dire les attributs graphiques qui permettent de représenter la hauteur.

2.3. Hauteurs spectrales et fréquence

À divers endroits de l'application, les hauteurs sont aussi exprimées en hertz. La hauteur en hertz n'est pas stockée, elle est simplement déduite de la hauteur de la note (sur base d'un la à 440 hz). De ce fait, on ne peut pas fixer la hauteur d'une note sur n'importe quelle valeur en hertz, on ne peut la fixer que sur une fréquence correspondant à une hauteur musicale. L'approximation dépendra du mode microtonal choisi. Pour des calculs spectraux, l'utilisation du mode microtonal le mieux défini est donc souhaitable. Divers outils de traitement ou de génération permettent aussi de manier la hauteur sous forme d'une valeur harmonique se référant à une fondamentale. Il est ainsi possible de faire de la « transposition spectrale » (addition d'un certain nombre de degrés harmoniques aux hauteurs existantes), ou de générer des dessins mélodiques sous forme de suites d'harmoniques. Ces calculs sont relatifs à la fondamentale choisie par l'utilisateur, un choix qui sera évidemment déterminant pour le résultat.

2.4. Hauteurs modales et échelles

Une échelle est un « choix de sons disposés par fréquence croissante ou décroissante. À la différence du mode, l'échelle n'établit aucune hiérarchie entre les sons » (Hakim et Dufourcet 1995) [9]. L'échelle est donc une couche d'abstraction supplémentaire à la notion de « mode ». Sequenza propose un outil très complet qui permet de créer et de manipuler des échelles « hors temps ». Les échelles sont présentées comme allant du plus grave au plus aigu mais elles ne doivent pas nécessairement être constituées d'une suite de hauteurs triées : le dessin peut être constitué de segments montants et descendants, par exemple à la manière de l'« ultrapolation » ou de l'« infrapolation » décrites par Slonimsky (1947) [10]. Ce type d'échelle n'est pas conseillé pour un traitement faisant appel à l'approximation, mais peut être intéressant quand le traitement visé travaille sur base de l'indexation, c'est le cas de la « projection modale » (projection des index d'un dessin mélodique d'une échelle dans une autre) ou de la « transposition modale » (addition d'index dans une échelle donnée).

2.4.1. Modes

Un mode est une suite d'intervalles (au minimum un) qui se répète en se transposant selon la somme de ses intervalles, généralement l'octave mais pas nécessairement, et qui dispose d'un degré fondamental. Les échelles qui répondent à cette définition, par exemple le mode diatonique, sont développées dans Sequenza comme couvrant tout l'ambitus de l'instrument.
Si l'intervalle entre deux degrés fondamentaux du mode est autre que l'octave, on parle alors de mode « non-octaviant ». De nombreux modes de ce type ont été décrits par Slonimsky (1947) [10] et sont très simplement réalisables dans Sequenza.

2.4.2. Échelles fixes

Dans Sequenza, une « échelle fixe » est constituée d'une suite de notes indéterminées, possiblement sans aucun principe de répétition. À la différence du mode, elle ne se répète pas et ne couvre donc pas nécessairement tout l'ambitus de l'instrument.
On peut utiliser les échelles fixes à des fins compositionnelles, mais aussi comme aide à l'introduction de données, par exemple pour ne laisser écrire dans un jeu de portées que les hauteurs incluses dans un set de percussions donné (voir 2.4.5).

2.4.3. Échelle des harmoniques

Dans Sequenza, l'échelle des harmoniques est représentée sous forme d'une échelle fixe en sixièmes de ton montant jusqu'à l'harmonique 64.

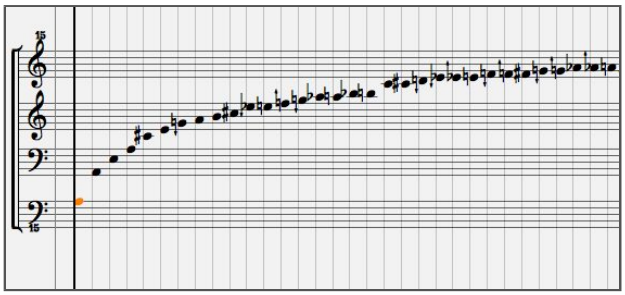


Figure 3. *Vue partielle de l'échelle des harmoniques dans l'éditeur d'échelle*

2.4.4. Stockage, parenté et identité

Tous ces types d'échelles sont représentables dans Sequenza et peuvent être sauvegardés sur le disque dur sous forme de bibliothèques indépendantes de la session. Chaque échelle possède son propre mode microtonal, elle impose aussi à la partition sa propre manière de représenter les altérations, ceci pouvant être ensuite adapté note à note par l'utilisateur. Chaque instrument possède une échelle (par défaut l'échelle diatonique) qui détermine la notation des altérations. Les séquences monoinstrumentales peuvent également contenir une ou plusieurs échelles réparties dans le temps, ce qui permet, notamment, à l'application de gérer les changements d'armure. Quand une échelle est attachée à la séquence, elle a priorité sur celle de l'instrument parent. Il est possible de transposer une échelle, ou de modifier sa fondamentale, ainsi pour exprimer les 7 modes diatoniques dans les 12 tonalités chromatiques, on n'a en fait besoin que d'une échelle dont on adaptera deux paramètres : la transposition et le degré fondamental.

2.4.5. Usage

De même que l'application utilise le magnétisme rythmique pour aider à l'introduction des rythmes (voir point 3.2), les échelles de Sequenza lui permettent d'opérer un magnétisme « harmonique », évoluant potentiellement dans le temps, qui peut être plus ou moins contraignant. Il peut ne servir qu'à aider l'introduction de notes dans la partition, ou à leur transposition interactive, mais il peut aussi magnétiser de manière systématique interdisant de fait l'existence de toute note n'appartenant pas à l'échelle dans la séquence. Ce magnétisme étant « non-destructif », il permet de tester l'alignement avec des échelles différentes : le calcul repart toujours des hauteurs originales qui sont conservées intactes - et qui peuvent d'ailleurs être rétablies à tout moment en désactivant simplement le magnétisme. Les échelles peuvent aussi servir à la sélection (ne sélectionner que les notes appartenant à une échelle données par exemple), au traitement ou encore à la génération de matériaux musicaux. Dans ce cas, les hauteurs, dont on a déjà vu qu'elles pouvaient être exprimées comme des notes ou des harmoniques, sont exprimées comme les degrés (les index) d'une échelle. Le script de génération produira dans ce cas un résultat dépendant du contexte, c'est-à-dire de l'échelle courante.

3. GESTION DU TEMPS

3.1. Mode proportionnel

Conservant là une filiation avec *Open Music* (OM 6.5 Score Objects) [11], Sequenza s'organise essentiellement en deux modes de représentation de la partition, l'un plutôt destinée au travail sur la modélisation sonore et sur la composition et l'autre sur le rendu graphique et la mise en page. Dans le mode « proportionnel » (appelé « notation

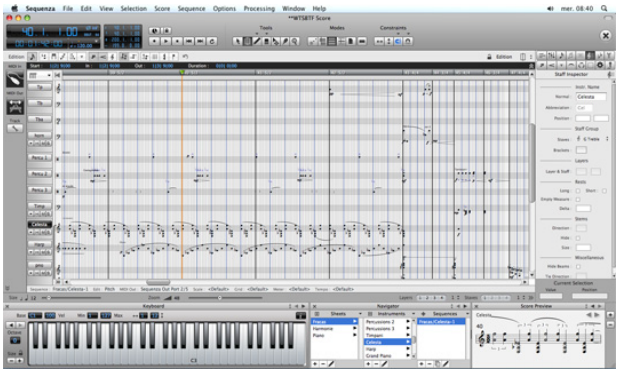


Figure 4. *Mode de représentation proportionnel*

MIDI » dans l'application), le temps est représenté de manière linéaire, un certain nombre de pixels équivalant à une certaine durée musicale. Deux rondes sont donc quatre fois plus espacées l'une de l'autre que deux noires. Dans ce mode, l'iconographie classique intervenant dans la notation du rythme n'est pas utilisée : toutes les notes sont noires, elles n'ont ni hampe ni ligature. Leur longueur est représentée par un trait de couleur dans une courbe dédiée. Lors du playback, le curseur de la tête de lecture se déplace donc à vitesse constante, au moins à tempo égal.
Des discontinuités peuvent toutefois être introduites dans cet espace proportionnel pour ménager un espace graphique à certains signes qui en ont besoin : tout particulièrement les clés et les armures qui interviennent en cours de séquence. Ces espaces-là n'ont aucune durée temporelle : quand le curseur arrive à un endroit de ce genre, il le « saute » purement et simplement.
Ce mode offre comme avantages, la simplicité, la vitesse d'exécution et la possibilité de représenter des longueurs temporelles importantes en une seule vue, permettant ainsi une appréhension visuelle des éléments formels.

3.2. Grilles

Pour découper le temps et aider à l'introduction de données rythmiques dans la séquence, Sequenza utilise le principe de la grille magnétique. C'est un principe bien connu, mais qui est ici poussé à l'extrême.
Il existe dans Sequenza trois types de grilles, chacune étant entièrement configurable par l'utilisateur. On peut librement afficher ou masquer ces grilles, dans la règle ou dans la partition.

3.2.1. Mètre

La grille métrique exprime les signatures temporelles. Cette grille peut être remise en forme de diverses manières : édition interactive, menu édition (copier coller, etc.), éditeur et inspecteur spécialisés. Cette grille n'a aucune incidence sur la manière dont le matériel musical est organisé et stocké, on peut changer le mètre sans aucune incidence pour la séquence : à ce stade, le mètre n'est qu'un artifice graphique nous permettant de diviser le temps.

3.2.2. Grille rythmique

Quoique toutes les grilles soient potentiellement magnétiques, la grille rythmique est certainement la plus importante en termes de notation du rythme. En théorie, elle est constituée de valeurs plus petites que celle de la grille métrique (mais il n'y a là aucune obligation) que l'on dispose afin de permettre l'introduction de valeurs rythmiques rationnelles. La grille est d'abord constituée d'un simple pas que l'on peut librement définir (1/16 par défaut). On peut ensuite choisir une autre valeur (n'importe quelle valeur rationnelle) pour un segment donné de la ligne de temps et personnaliser ainsi la grille peu à peu. On peut aussi diviser un segment : par exemple, sélectionner une région de la longueur d'une noire et la diviser en trois – en pressant 3 – pour écrire ensuite un triolet.
Les valeurs de la grille sont utiles à production ou à la transformation de données, mais une fois les notes positionnées, on peut modifier la grille sans altérer la notation rythmique de la séquence. Tout comme la grille métrique, la grille rythmique dispose de nombreux outils d'édition.

3.2.3. Grille polyrythmique

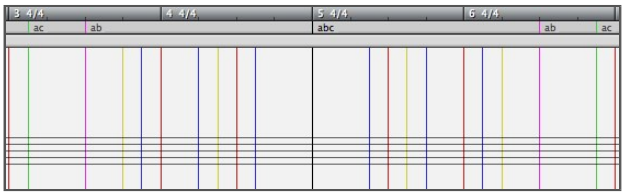


Figure 5. *Un segment de grille polyrythmique (la mise en phase est au centre)*

La grille polyrythmique offre la possibilité de déployer dans le temps un maximum de trois valeurs rythmiques indépendantes des deux autres grilles et, singulièrement, de la grille métrique. Ceci permet d'écrire des éléments temporels périodiques sortant du cadre de la mesure comme c'est souvent le cas dans l'écriture polyrythmique. Les trois valeurs sont librement définies, on peut aussi n'en utiliser qu'une ou deux. Il est également possible de déterminer où se situe leur première « mise en phase » (par défaut, à l'onset 0).
Graphiquement, chaque valeur est symbolisée par une couleur primaire, les points de rencontre entre deux valeurs sont symbolisés par la couleur complémentaire résultante, les points de rencontre entre les trois valeurs sont symbolisés par la couleur noire (voir figure 5).

3.2.4. Marqueurs temporels

À ces 3 types de grille s'ajoutent encore les marqueurs temporels qui permettent de marquer des points temporels non périodiques.

3.3. Tempo

Dans Sequenza, le tempo couvre une plage de 8 bits déployée entre 16 et 271. Afin de permettre la gestion de tempos plus rapides, un facteur multiplicateur est utilisé qui permet de multiplier le tempo de base par 2, 4 ou 8 permettant de fait l'utilisation d'un tempo montant jusqu'à 2168… Ceci nous mène d'ailleurs aux portes de la musique électronique, mais c'est un point que je ne développerai pas ici.
Les variations de tempo sont traitées dans des courbes qui sont représentées de la même manière que les courbes MIDI (volume, panoramique, etc.). Ces courbes permettent l'expression, au choix, d'une série de valeurs discrètes (comme dans le langage MIDI) ou d'une série de points reliés par des diagonales dont les valeurs discrètes sont calculées à la volée.

3.3.1. Processeurs liés au tempo

Le traitement de musique à tempos multiples superposés est une demande qui pose un véritable problème de cohérence quand on souhaite maintenir un lien avec une partition générale dont le principe est celui d'un tempo unique, possiblement battu par un chef. Sequenza propose plusieurs outils permettant de traiter le tempo indépendamment d'une piste à une autre sans pour autant briser l'unité temporelle nécessaire à la partition et au moteur temps réel.
Le processeur de « compression-extension temporelle » est un principe bien connu qui permet ici de traiter la plupart des objets de la partition, y compris ceux qui sont purement graphiques (comme les expressions). La façon dont le facteur de compression est contrôlé correspond particulièrement à la nécessité d'une musique polyrythmique à tempo multiple.
La « consolidation de tempo » permet d'appliquer une suite de changements de tempo dessinée dans une courbe aux valeurs rythmiques des accords de sorte que le rendu sonore reste identique une fois la courbe de tempo effacée (ceci induit une réorganisation des accords dans le temps). Il est ainsi possible de superposer plusieurs fluctuations de tempo complètement différentes dans des pistes distinctes (par exemple un accelerando et un rallentendo superposés et synchronisés).

3.4. Mode de notation classique

Comme présenté à la figure 6, le second mode de représentation de Sequenza est un mode de notation classique

présenté au format page.
Une des originalités de Sequenza est sa manière d'utiliser le système de stockage du temps musical pour générer une partition classique automatiquement. Il s'agit certainement du point sur lequel le travail de recherche et développement a été le plus important.

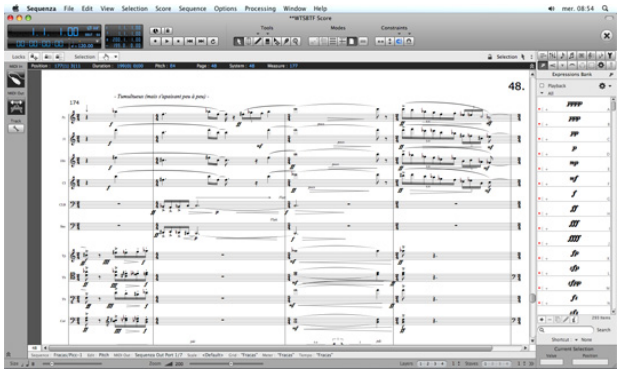


Figure 6. *Mode de représentation classique*

3.4.1. Codage de la position temporelle

Les positions temporelles des accords et des grilles de Sequenza sont stockées sous forme de fractions alors que, dans le langage MIDI, les valeurs temporelles sont stockées sous forme de « ticks » (The MIDI File Format, PPQN Clock) [12].
Certains logiciels codent leurs fichiers MIDI avec un nombre de ticks de 1024 par noire (2^{10}), ce qui est un choix cohérent d'un point de vue informatique, mais assez peu d'un point de vue musical car ce chiffre n'a que 2 comme facteur : un simple triolet ne peut être codé sans approximation. Historiquement, les constructeurs ont donc généralement privilégié des nombres qui ont au moins deux facteurs : 2 et 3. Ainsi 96, 192, 384 ou 768 ont été des standards selon les époques. « Protools » utilise 960, un nombre qui présente l'avantage d'être également divisible par 5.
Quoi qu'il en soit, un programme qui doit pouvoir déduire une notation rythmique musicale incluant des groupes complexes sur base d'onsets temporels sans erreur d'approximation ne peut pas se baser sur un système de codage en ticks. Seul le codage d'un numérateur et d'un dénominateur comme deux nombres entiers permet de représenter des valeurs rythmiques sans aucune erreur d'approximation. C'est pourquoi Sequenza a fait ce choix.

3.4.2. Conversion en notation classique

Il n'est pas possible de donner ici une explication détaillée de l'algorithme qui permet à Sequenza de transcrire automatiquement une notation simplement proportionnelle en notation métrique. La complexité de cette procédure tient notamment à la multiplicité des interprétations possibles souvent dépendantes de la culture et de la tradition. Voici toutefois un exemple simple :

Imaginons une valeur temporelle rationnelle placée quelque part dans la ligne de temps. Il s'agira vraisemblablement d'un nombre difficilement lisible, mais on peut très facilement le simplifier en le rendant relatif au début de la mesure. Imaginons que ce nombre devienne alors 9/28.

Le dénominateur 28 étant égal à $2^2 \cdot 7$, il en ressort que 1/28 est la valeur de base d'un septolet de doubles-croches.

Le quotient du numérateur $9 : 7 = 1$ indique que ce septolet commence sur la seconde noire de la mesure et le reste ($9 \bmod 7 = 2$) indique que la valeur se situe sur le troisième pas du septolet.

L'analyse du numérateur nous indique toutefois que 9/28 pourrait aussi être le quatrième pas d'un groupe de 7 croches divisant une blanche pointée commençant au début de la mesure.

Il faudra donc vérifier ces extrapolations à la lumière du contexte, des autres membres du groupe, et des marqueurs d'unités de la signature temporelle. Ceci se fera dans le flux de l'analyse séquentielle des valeurs rythmiques de la mesure (couche polyphonique par couche polyphonique), en réévaluant les extrapolations faites au fur et à mesure que l'analyse affine, confirme ou infirme les précédentes hypothèses.
Si l'analyse débouche sur une impossibilité, due à une trop grande complexité ou à une valeur irrationnelle, le groupe en cours est simplement aligné graphiquement sur les valeurs binaires les plus proches.

Le bon ajustement de cet algorithme nécessite la recherche d'une limite raisonnable à la complexité rythmique tolérée. En effet, l'ouverture à un niveau de complexité trop important peut fragiliser l'efficacité de l'interprétation de données simples ce qui tendrait à rendre l'application obscure dans ses résultats.
Sequenza limite donc ses recherches à tous les facteurs premiers inférieurs à 32.

4. RENDU GRAPHIQUE ET RENDU SONORE

Toute la problématique de la relation entre rendus graphique et sonore commence quand on rencontre des « gestes » musicaux qui ne s'expriment pas de la même manière dans les deux registres. Un trille qui s'écrit comme une simple valeur longue surmontée d'un signe dans la partition et qui se joue comme une série de valeurs brèves alternées. Un accord brisé qui s'écrit aligné verticalement au côté d'un signe graphique signalant que l'accord doit être « arpégé » alors que le rendu sonore nécessite l'écriture d'une série de notes non alignées. On pourrait multiplier les exemples, ils sont nombreux.
Face à cette problématique, Sequenza offre une gamme variée d'outils qui mettent en œuvre différentes stratégies. Leur combinaison permet de trouver une réponse adéquate à la plupart des problèmes posés. La complexité induite par ces artifices peut être compensée par une automatisa-

tion partielle des opérations.

4.1. Durée et durée graphique

Au sens du langage MIDI, la durée est simplement la distance en ticks qui sépare le message « Note On » (enfoncement de la touche) du message « Note Off » (relâchement de la touche). Chaque note possède donc sa propre durée.
D'une autre coté, au sens de la représentation dans la partition, chaque accord possède une durée qui est exprimée sous forme d'une valeur rationnelle binaire, possiblement contractée par le fait qu'elle se trouve dans un groupe irrégulier, possiblement allongée par le fait qu'elle soit pointée, et possiblement brisée en plusieurs valeurs rythmiques liées.
Ceci montre à quel point ces deux notions sont distinctes et à quel point le passage de l'une à l'autre demande un travail d'interprétation important.
Sequenza propose deux valeurs distinctes pour la durée. La première est l'équivalent de la durée MIDI, exprimée en ticks et stockée note à note : c'est la durée audible commune à tout séquenceur MIDI. La seconde, moins courante, est appelée « durée graphique » et est stockée dans l'accord. Cette valeur sert essentiellement à préciser la durée que l'on souhaite donner à un accord quand il est converti en notation classique.

4.2. Positionnement uniquement audible

Comme dans *Open Music*, chaque note possède une position temporelle purement sonore appelée « offset » qui est relative à son accord parent. Cette valeur peut changer le positionnement audible de la note de maximum une noire dans les deux directions du temps. Ceci permet, par exemple, d'écrire un accord qui est brisé du point de vue audible, mais qui reste aligné graphiquement.
L'offset est aussi utilisée pour offrir la possibilité d'aligner un phrasé rythmique (par exemple enregistré en temps réel) sur la grille de manière seulement graphique : cette valeur garde alors mémoire des deltas entre la « performance » et les valeurs alignées.

4.3. Accords invisibles et accords inaudibles

Les accords peuvent être déclarés comme étant inaudibles ou invisibles. Les accords invisibles restent visibles (quoique semi transparents, voir figure 7) en mode proportionnel, mais ne le sont plus du tout en mode de notation classique. De ce fait, ils sont exclus du calcul d'espacement et de mise en page. Ils restent cependant toujours audibles. Les accords inaudibles se transforment quant à eux en objets purement graphiques. Il est ainsi possible de dissocier à l'extrême ce qui est écrit de ce qui est joué.

La figure 7 donnent un exemple de l'utilisation des accords invisibles et inaudibles pour construire une geste musical dont les représentations sonore et graphique diffèrent largement. Les accords en vert pâle sont invisibles en notation classique. Au contraire, les accords visibles en notation classique sont inaudibles.

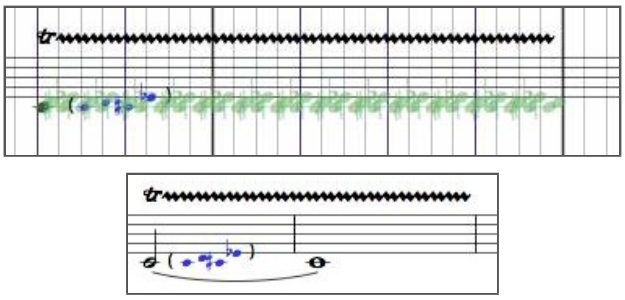


Figure 7. *Un trille complexe en mode proportionnel, puis le résultat graphique en notation classique*

4.4. Couches polyphoniques

Ce concept - qui s'apparente à celui de *calque* dans un éditeur graphique - est utilisé de par la plupart des logiciels de la famille des éditeurs de partitions. Dans Sequenza, les couches polyphoniques permettent d'écrire dans quatre couches indépendantes par portée, et il est possible de les régler pour qu'elles soient invisibles ou inaudibles. Il y a donc là une solution plus générale à la problématique abordée au point précédent.

5. ARCHITECTURE

Comme utilisateur d'informatique musicale, j'ai toujours trouvé frustrante l'organisation de la grande majorité des applications autour d'une ligne de temps unique ne comprenant de surcroît qu'un jeu de méta-événements (mètre et tempo) [3] . Cette conception ne me semble pas en phase avec le travail du compositeur qui est souvent constitué de brouillons, de recherches et d'essais multiples, de versions diverses, le tout s'ajoutant naturellement à la partition principale et constituant avec elle un ensemble intellectuel homogène. C'est pourquoi Sequenza est organisé très différemment.

5.1. Instrument et feuille

Au traditionnel empilement d'instruments, Sequenza ajoute une dimension : la « Feuille » (Sheet). Ce terme fait référence au travail sur papier dans lequel le compositeur travaille sur une partition générale, mais peut aussi avoir d'autres matériaux complémentaires sur des feuilles distinctes.

Autant l'instrument est un objet « horizontal » destiné à être superposé à d'autres dans la partition, autant la feuille peut être vue comme un objet « vertical » (parce qu'il « traverse » tous les instruments) destiné à en côtoyer

3 . Le terme de « méta-événement » fait ici référence à la norme MIDI et à son format de fichier. Tout événement qui n'est pas spécifiquement lié à une piste, mais concerne l'ensemble du morceau, est nommé « meta event » et doit être placé dans la piste zéro. On y trouve tout particulièrement les changements de tempo et les signatures temporelles.

d'autres dans une relation beaucoup moins déterminée. Les feuilles ne sont pas des « parties » car cela supposerait que ces objets soient destinés à s'enchaîner dans le temps, ce qui n'est pas le cas. Les feuilles sont plutôt le lieu où faire des essais divers, où décliner des versions, où construire une série de mouvements faisant partie d'un même ensemble, mais sans que cet ensemble ait nécessairement une organisation temporelle déterminée.

5.2. Cellule, séquences et héritage

Pour aider à la compréhension de ce qu'est une cellule, on peut prendre l'exemple d'un tableur dont les instruments seraient les rangées et les feuilles les colonnes [4].

De la jonction entre une feuille et un instrument naît une cellule dont la seule fonction est d'être un réservoir de séquences (au minimum une). Chaque cellule, et chaque séquence-enfant, a donc deux parents, un instrument et une feuille.

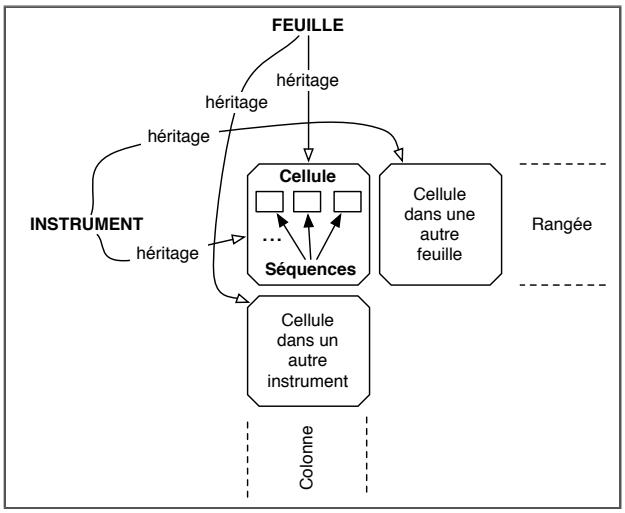


Figure 8. *Feuille, Instrument, Cellules et Séquences*

Autant l'instrument est le conteneur naturel des objets liés à la piste (système de portées, destination MIDI, mode microtonal, échelle, etc.), autant la feuille est le conteneur naturel des méta-événements [3] (grilles, mètre, tempo, marqueurs temporels, etc.) qui affectent toutes les pistes.

Par défaut, chaque séquence hérite des propriétés de ses deux parents, mais peut ensuite « autonomiser » certaines d'entre elles en se les attachant directement. Ce système d'héritage donne à Sequenza une grande souplesse de fonctionnement, partant d'une situation simple où les séquences partagent toutes les propriétés de leurs parents et évoluant, quand nécessaire, vers une situation plus sophistiquée où chaque séquence possède ses propres propriétés.

4 . Si, dans ce tableur, l'axe des y représente la superposition des instrument dans la partition, l'axe des x ne représente pas le temps : c'est plutôt une abstraction liée à l'organisation des données (comme, par exemple, un onglet dans un navigateur).

5.3. Navigation

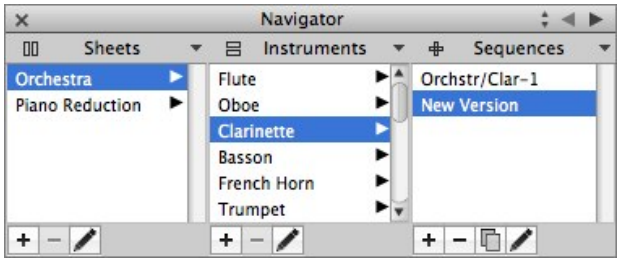


Figure 9. *Navigation dans la structure d'une session*

Le navigateur de Sequenza, présenté à la figure 9 permet la navigation dans la structure de la session. Il adopte une présentation proche de celle d'une arborescence de fichier. La cellule courante est représentée par le réservoir de séquences qui se trouve à la droite de la vue.

5.4. Feuille et multiséquence

Chaque feuille contient une ou plusieurs partitions poly-instrumentales appelée multiséquence. Comme la séquence simple, la multiséquence hérite ses méta-événements de la feuille parente et peut aussi s'attacher en propre un ou plusieurs types de méta-événements.

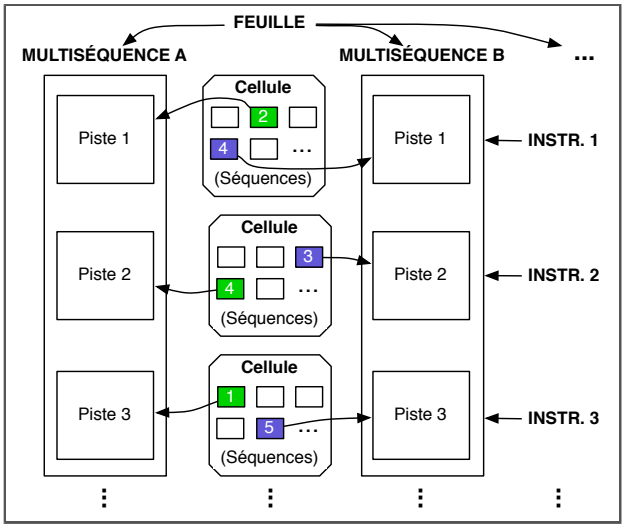


Figure 10. *Organisation des multiséquences dans une feuille : De même que chaque cellule comporte un nombre illimité de séquences, chaque feuille comporte un nombre illimité de multi-séquences (par défaut une seule)*

Une multiséquence contient N séquences, une pour chaque piste. Comme on le voit à la figure 10, dans chaque multiséquence la piste peut utiliser l'une des séquences disponibles dans le réservoir de séquences que constitue la cellule.

5.5. Feuille et multi-instrumentation

Afin de ne pas être limité par le caractère « rectangulaire » du tableur évoqué au point 5.2, chaque

feuille peut ne pas utiliser tous les instruments disponibles dans la session. Il est ainsi possible - quoique la session ne comporte qu'un seul jeu d'instruments - d'avoir des effectifs instrumentaux partiellement ou même totalement différents dans les différentes feuilles.
Par exemple, pour réaliser une pièce pour orchestre dans une feuille et sa réduction pour piano dans une autre, il suffit que la session comporte un effectif instrumental constitué d'un orchestre plus un piano. Le piano sera désactivé dans la première feuille, et l'orchestre sera désactivé dans la seconde feuille.

5.6. Parties individuelles

Enfin, pour travailler sur une partie individuelle (par exemple pour la production d'un matériel orchestral), il suffit de passer du mode de représentation poly-instrumental (où la multiséquence est l'objet représenté dans la partition) au mode mono-instrumental (où la séquence est l'objet représenté dans la partition) [5].

6. PARTAGE

Sequenza se veut le plus ouvert possible au partage de ses données afin de pouvoir s'intégrer harmonieusement à un ensemble logiciel. Voici un rapide tour d'horizon des moyens par lesquels Sequenza communique avec le monde extérieur :

MIDI File (import et export des données MIDI)
xmlMusic (export graphique vers un éditeur de partition)
PDF (export graphique)
Wave et Aiff (export audio)
Mov, Mp4, Avi, M4v, Dv (import multimédia)
SDIF (import d'analyses sonores spectrographiques)
SF2 (import de bibliothèques sonores)

Figure 11. *Travail avec fichier vidéo au format QuickTime*

5 . Le concept de mode mono-instrumental ou poly-instrumental est totalement indépendant du concept de feuille : On peut passer d'un mode à l'autre indépendamment du fait que l'on soit dans l'une ou l'autre feuille. Le mode poly-instrumental utilise toute la « colonne » tandis que le mode mono-instrumental n'utilise que l'une de ses cellules.

6.1. Ports de communication temps réel

Sequenza communique en temps réel au format MIDI et peut créer un nombre illimité de ports de sortie virtuels. Les données entrantes sont aiguillées vers le port actif (MIDI bypass) où elles peuvent aussi être enregistrées. Actuellement, le nombre de types d'événements MIDI interprétés est toutefois limité.
Sequenza dispose également d'un clavier virtuel sophistiqué qui offre des fonctionnalités originales pour la commande des données MIDI via le clavier alphanumérique.

6.2. Plug-ins

Sequenza héberge et commande tout type d'instrument virtuel compatible avec Mac OS X. Les données des plug-ins (AUPreset) sont préservées dans la session elle-même.

6.3. Scriptage

Sequenza possède un système de scriptage qui permet à l'utilisateur de générer ou de transformer de l'information dans la partition. Le langage de script utilisé est orienté vers la production de séries de valeurs qui représentent des hauteurs, des rythmes, des durées ou des dynamiques. L'organisation des scripts en modules et chaînes de modules ressemble assez à une forme de synthèse sonore dont le résultat serait, non pas un signal audio, mais de la musique écrite.
Les scripts sont organisés en « Patterns » qui peuvent être sauvegardés dans une banque et exportés dans la partition en cliquant simplement sur un bouton, ou en utilisant un mode spécifique de l'outil crayon.
Il est aussi possible d'importer de la musique dans l'éditeur, c'est-à-dire de convertir automatiquement un segment de partition en script, afin de pouvoir la manipuler.

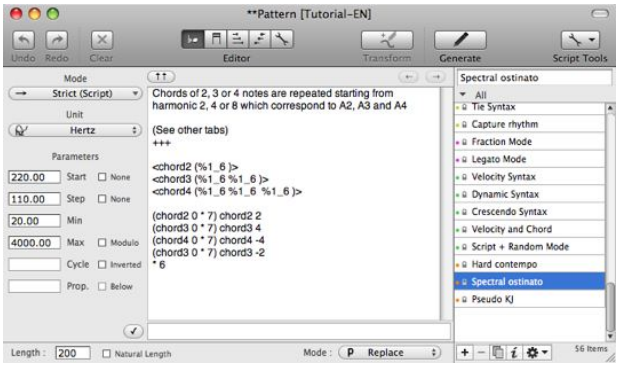


Figure 12. *La fenêtre de scriptage*

À moyen terme Sequenza devrait aussi pouvoir être contrôlé via *AppleScript*.

7. CONCLUSION

Sequenza tente d'apporter une contribution à tous ceux qui travaillent aujourd'hui à la reconstruction pour notre

musique, sinon d'un langage commun, au moins d'un espace conceptuel partagé, tout particulièrement en s'appuyant sur les techniques, si riches et si nombreuses, que nous a légué le XXe siècle. Il m'a toujours semblé que l'une des responsabilités de notre génération était de consolider les nombreux concepts élaborés durant ce siècle en les intégrant pleinement à une théorie musicale modernisée et en leur donnant toutes sortes de prolongations constructives.
C'est aussi à cette lumière qu'il faut lire le caractère parfois « vulgarisateur » du projet Sequenza. Tout en maniant des concepts parfois ardus de la théorie musicale classique, Sequenza se veut une porte ouverte vers ces savoirs pour ceux, nombreux, qui venant d'autres espaces culturels en sont trop souvent intimidés.
La plupart des fonctionnalités examinées ici illustrent cette volonté de débarrasser l'utilisateur d'opérations souvent trop fastidieuses - voire décourageantes - dès que l'on touche à ces questions que l'industrie considère sans doute comme marginales : groupes complexes, microtonalité, modes non-octaviants, polyrythmes, etc.
Pour le compositeur débutant, le fait que ces manipulations soient plus simples est une opportunité de s'ouvrir à ces concepts, d'en comprendre l'intérêt créatif, et de les intégrer plus aisément à son vocabulaire.
Pour le compositeur aguerri, cette mise sur un pied d'égalité des différentes techniques d'écriture, récentes ou plus anciennes, permet une clarification du langage et un gain de temps dans l'exécution des tâches. Ainsi, l'attention peut être plus largement orientée vers la dimension créative du travail de composition.

8. REMERCIEMENTS

Je souhaite remercier ici tous ceux qui m'ont aidé à porter le projet Sequenza :

L'équipe de développement d'*Open Music* dont travail m'a inspiré et aidé, et tout particulièrement Gérard Assayag, qui m'a donné de nombreux conseils au cours de la première phase de développement.

L'équipe d'ingénieurs de Lispworks ltd, et tout particulièrement Martin Simmons, et la communauté d'utilisateurs de *LispWorks*, qui m'ont apporté un soutien précieux au fil de ces années.

L'équipe d'ingénieurs du *CETIC*, et tout particulièrement Christophe Ponsard, qui m'a véritablement ouvert l'esprit en matière d'architecture logicielle au cours de la dernière phase de développement. Le *CETIC* héberge aujourd'hui gracieusement les codes de Sequenza dans sa forge.

Tous les bêta-testeurs, compositeurs et amis qui m'ont aidé à réévaluer, à améliorer et à corriger Sequenza, lors de ses phases alpha et bêta, notamment Stéphane Roumieux, Maya Tell-Nohet, ainsi que les étudiants de ma classe de *Composition en Musiques Appliquées et Interactives* du Conservatoire royal de Mons qui ont bien voulu s'impliquer dans cette phase de mise au point.

Je remercie enfin Irène Deliège et Jean-Luc Fafchamps pour leur relecture amicale et leurs conseils avisés.

9. REFERENCES

[1] LispWorks Ltd. *Objective-C and Cocoa User Guide and Reference Manual*, 2008.

[2] MIDI Manufacturers Association *MIDI 1.0 Specification*, 1982.

[3] Apple Inc. *Audio and MIDI on Mac OS X* (p36) 2001.

[4] OM 6.5 *User Manual, section MIDI 3.e Microintervals*, IRCAM. <http ://support.ircam.fr/forum-ol-doc/om/om6-manual/co/Microintervals.html>

[5] MIDI Manufacturers Association *MIDI Manufacturers Investigate HD Protocol*, 2008-2012.

[6] MIDI Manufacturers Association *MIDI Tuning Messages*, 2008-2012, 1995-2012. <http ://www.midi.org/techspecs/midituning.php>

[7] Letz S., Fober D., Orlarey Y. *Modèles de structures tonales dans Elody*. GRAME / JIM, 1995.

[8] Rasch R. *The Unification of Tonal System, or About The Circle of Fifths* (p 320 & 326), Journal of New Music Research, 2000.

[9] Hakim N. et Dufourcet M-B. *Guide Pratique d'Analyse Musicale* (p 110). Éditions Combre, Paris, 1995.

[10] Slonimsky N. *Thesaurus of Scales and Melodic Patterns* (p ii), Amsco Publication, New York, 1947.

[11] OM 6.5 *User Manual, section Score Objects 1.2 Time Representation*, IRCAM. <http ://support.ircam.fr/forum-ol-doc/om/om6-manual/co/Score-Objects-Intro.html#aeNe6>

[12] MIDI Manufacturers Association *MIDI File Specification - PPQN Clock*, 1988.

GESTURES, EVENTS AND SYMBOLS IN THE *BACH* ENVIRONMENT

Andrea Agostini
Freelance composer

Daniele Ghisi
Composer - Casa de Velázquez

RÉSUMÉ

Environments for computer-aided composition (CAC for short), allowing generation and transformation of symbolic musical data, are usually counterposed to real-time environments or sequencers. The counterposition is deeply methodological: in traditional CAC environments interface changes have no effect until a certain 'refresh' operation is performed, whereas real-time environments immediately react to user input. We can call the former approach *speculative*, as it is associated with a careful, preemptive formalization of musical ideas, and the latter *performative*, as it is intrinsically apt to deal with extemporariness and able to react to gestures - in the broadest sense of the term - as soon as they are performed. This distinction is by no means natural, since interactivity is an essential performative aspect of the musical discovery process. The reunification of the performative and speculative aspects is obtained via a MaxMSP library named *bach: automatic composer's helper* [1], which is a set of tools for symbolic processing and graphical music representation, designed to take advantage of MaxMSP's facilities for sound processing, real-time interaction and graphical programming. The *bach* environment is capable to extend the typically speculative paradigm of CAC, embracing the performative one as well, thus becoming a very natural tool for composers both during their own discovery process and in concert situations: external gestures, codified in input as events, handle any parameter change within the symbolic framework, immediately affecting a resulting real-time score.

1. INTRODUCTION

It is not surprising that, since the advent of computers, there has been great interest on how to take advantage of the superior precision, speed and power of electronic computers in music-related activities. The probably best-known (and commercially successful) direction has proven being the generation and transformation of sound. In recent years, inexpensive personal computers (and lately even top-end mobile phones) have gained the ability to perform professional-quality audio transformation and generation in real-time. On the other hand, several systems have been developed to process symbolic data rather than acoustic ones - 'notes' rather than 'sounds'. These systems can be roughly divided into tools for computer-assisted music engraving (Finale, Sibelius, Lilypond...) and tools for computer-aided composition (CAC for short, allowing generation and transformation of symbolic musical data, like OpenMusic, PWGL, Common Music...). Moreover, at least two graphical programming environments, the closely related Max and PD, have MIDI control and sound generation and transformation among their main focuses - but at the same time they are capable to deal with arbitrary set of data, input/output devices and video. Indeed, the boundaries between all these categories are extremely fuzzy: music engraving systems often allow non-trivial data processing; some sequencers also provide high-quality graphical representation of musical scores and sound treatment; modern CAC environments include tools for sound synthesis and transformation. It should though be remarked that Max and PureData have very crude native support for sequencing, and no support whatsoever for symbolic musical notation. Moreover, an orthogonal distinction should be made between real-time systems, which 'immediately' react to interface actions (such as Finale, Max, ProTools...) and non-real-time systems, where these actions have no effect until a certain 'refresh' operation is performed (such as Lilypond, OpenMusic, PWGL). The latter is the case of typical CAC environments; yet, this is by no means natural: there is no deep reason why symbolic processing should not be performed in real-time, the only reason being what we could label a 'technological anachronism'. In fact, advanced symbolic computation and musical representation can easily become very costly in terms of processing power, and personal computers could not stand its computational weight until a few years ago. This situation has established a traditional separation that, although still lingering on, is no longer justified, since interactivity is an essential performative aspect of the musical discovery process, allowing any input gesture to immediately affect a given score.

2. GESTURES, EVENTS, SYMBOLS

Before getting to the core of the article, we have to briefly investigate the meaning of the word 'gesture' - or, more appropriately, the broader meaning of a family of words derived from the Latin 'gestus', in turn the past participle of the verb 'gerere', in several modern European languages: the French *geste*, the German *Geste*, the Italian and Spanish *gesto*... While there is a broad area of superposition in the meanings of all these words, some specific connotations are more apparent in some languages than in others. Generally, we will use the English 'gesture' in its broader sense, encompassing all the possible different

[1] http://www.bachproject.net

shades of connotations.

We talk about hand gestures, musical gestures, demonstrative gestures, hardly separating the literal meaning and the metaphorical usage of the word. In any case, it appears to us that roughly all gestures have at least one of these two 'features' inside their meaning:

- temporality: gestures are seen or interpreted in some temporal development. This can either involve actual movements in space (a finger hitting a key on a piano, or clicking on a mouse), or acoustic time-processes (a violin crescendo), or any synaesthetic extension (a black line which gets thicker on a white background, interpreted as the operation in time of the painter who drew it);

- extrapolation: a gesture is an identified portion of a whole, which stands out for some particular reason. This extrapolation is *always* an operation of meaning: gestures comes out by according a particular meaning with respect to the whole: a straight white line on a black canvas is a gesture since it stands out from the background; the gesture of a finger hitting and then releasing a piano key is an extrapolation from the all the movements of the finger in time; an orchestral fortissimo in a pianissimo situation is a gesture inasmuch as it stands out from it; even an entire piece, like Beethoven's *Grosse Fuge* may be a gesture in itself, if seen inside the whole Beethoven's production, or inside the entire history of music.

These two characterizations still reflect two etymological aspect within the Latin *gerere*: temporal behaviour (to bring, to pass, to spend...) and event-centering (to do, to produce, to show...).

Focusing on the musical case, we are not only interested in all the physical gestures (the violinist moving the bow, the pianist hitting the key, the composer clicking with the mouse in the patch), but also the musical gestures, conceived as meaningful properties of a score, an interpretation, an improvisation. Indeed, a real-time CAC involving symbolic musical gestures is not only possible but also achieved by the *bach* environment.

3. REAL-TIME GESTURE HANDLING AND REPRESENTATION WITH *BACH*

The creation and modification of a musical score is not an out-of-time activity, but it follows the composer's discovery process and develops accordingly. Persuaded that a composer-friendly CAC environment needs to cope with this fact, it was our priority that any data could be handled in real-time: composers working with symbolic gestures may need the machine to quickly adapt to new configurations, as they change some parameters.

Of course, *bach* is not the first attempt to address this dichotomy. Several interesting projects have been developed, linking real-time environments to graphical representations of both classical and non-classical (and potentially non-musical) scores, including OpenTimeLine [2] [6], MGC [3] [10], INscore [4] [5]. In at least one case, namely MaxScore [5], this is augmented by a very sophisticated editing interface. With *bach*, as shown in [1], we have tried to achieve a coherent system explicitly designed for computer-assisted composition: the advanced editing interface is conceived to take advantage of the combination of interactive writing and algorithmic control of symbolic musical material. Thus the set of graphical editors is focused on the representation of the underlying structures of music, rather than their graphical nuances. Therefore *bach* is not a single editor object, rather a rich set of tools for symbolic processing and graphical musical representation; each element in the environment is designed to take advantage of Max's facilities for sound processing, real-time interaction and graphical-programming.

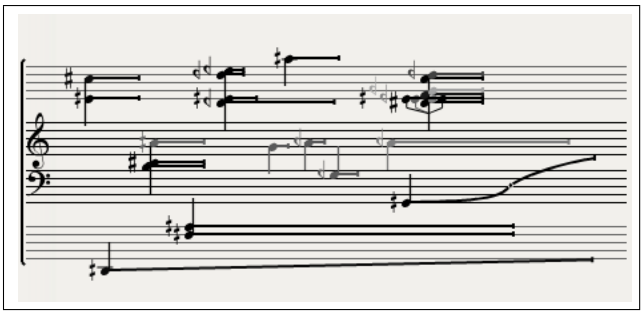


Figure 1. A simple *bach.roll*, representing a musical score in proportional notation

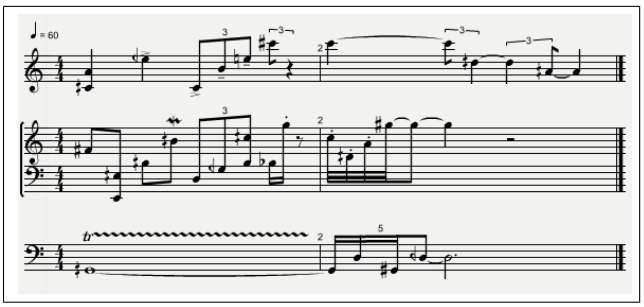


Figure 2. A simple *bach.score*, representing a traditional score (with measures, time signatures and tempi).

bach follows the general data-flow paradigm of Max: users build a sort of 'assembly chain' made of very specialized operators. In fact, any action feeding data into the 'assembly chain' can be considered as a specific event, happening at a certain moment in time. This applies particularly well to the typical performative usage of Max: events can be, for instance, actions performed on a MIDI controller, or in front of a camera, or sounds captured by a microphone - as well as mouse clicks and letters typed

[2] http://dh7.free.fr
[3] http://www.numediart.org/projects/12-2-mouse-gesture-composer
[4] http://inscore.sourceforge.net
[5] http://www.computermusicnotation.com

on an alphanumeric keyboard, of course. If the reaction to an event is immediate enough, we have the perception that the system is responding in real-time to our input. This leads to the idea that, if we assume that a gesture can be seen as a particularly structured sequence of punctual events, on a higher level of abstraction Max can indeed be controlled in real-time by gestures.

If we think at standard CAC environments (such as Patchwork, OpenMusic and PWGL), instead, we can observe that their programming paradigm is deeply different, as the entry of data does not trigger any immediate reaction: in these systems, an explicit evaluation command must be given to the machine in order to perform the desired operation. The difference is less subtle than it might appear, both on the conceptual and practical point of view. Of course, the evaluation command is just a key pressure, or a mouse click; but whereas this single action might not be very critical in a non-real-time context, it becomes crucial when synchronicity matters, or when the data flow comes, for instance, from a MIDI keyboard, or a microphone, or a video camera. The non-real-time approach is thus unable to properly keep track of a stream of incoming events, be they a sequence of keys pressed by a player or a series of instructions guiding the composer's thought.

It should be clear at this point that the choice of implementing a CAC environment within a real-time event-driven system leads to important consequences in the kind of scenarios it can be applied to. Firstly, if the composer's interface choices affect in real-time the symbolic result, the machine feedback is way more extensive and intuitive, allowing the user to handle and validate a much larger set of possibilities. In this sense, *bach* allows a sort of 'trial and error' approach to symbolic data, in a similar way as composers generally do while writing electronic music. But, most importantly, *bach* introduces the possibility of a gestural approach to symbolic generation of, and intervention upon, musical materials. Of course one has to strike a balance between the amount of feedback obtained from the machine and the amount of information that the composer himself can favorably process. As noticed in [1], *bach* is meant to close the gap between the 'performative' and 'speculative' aspects of tools for computer music, which has pushed the real-time and CAC communities apart.

As a final note on the subject, it is worth to underline that the real-time paradigm is a resource rather than an obligation: score changes are handled by the user, who is completely free to make them happen immediately or only after some sort of 'refresh' operation (as it would be the case in non-real-time environments). This means that, in principle, nothing prevents from using *bach* just like any other CAC environment, and there are cases (such as a fine rhythmic quantization) in which one is obliged to settle in the non-real-time paradigm, since the significant amount of time needed for performing a particular task might actually disrupts the immediateness of response to the user's actions.

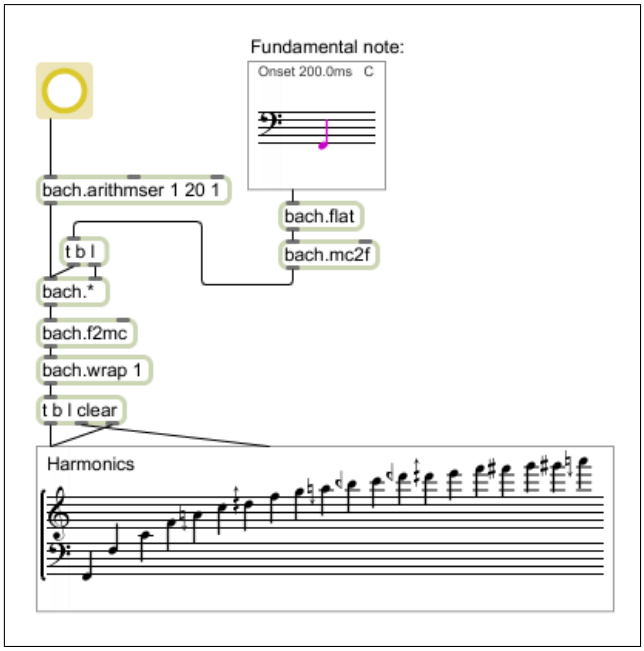


Figure 3. A simple patch calculating the first twenty harmonics of a given note. Either clicking the *button* or dragging the fundamental note will immediately update the result.

4. THE *BACH* ENVIRONMENT

4.1. The components of *bach*

As already stated, *bach* is a library of objects and abstractions for the software Max, the distinction between objects and abstractions concerning more the implementation than the actual usage of these modules. Both appear to the user as graphical widgets that can be linked to others in order to build the 'assembly chain' that will perform the desired operations. The difference lies in the fact that objects are written in C and - unless the source code is made available - behave like non-editable 'black boxes', whereas abstractions are combinations of objects and other abstractions, which users are always allowed to open and edit; the drawback for this flexibility is that abstractions are usually far less efficient than properly written objects, and some functionalities simply cannot be implemented without recurring to some C coding: this is particularly true when dealing with custom data types and advanced graphical operations, which is the case for a large portion of the *bach* environment. In Fig. 3 we show a sample patch containing some *bach* objects and abstractions.

In addition to that, great care has been posed in the documentation of *bach*. Every object and abstractions comes with a fully detailed help file; a general help patch can be recalled from the 'Extras' menu of Max; a growing set of tutorial patches provide a step-by-step introduction to most features of the system. Moreover, a user forum exists and a new set of video tutorials is currently under development.

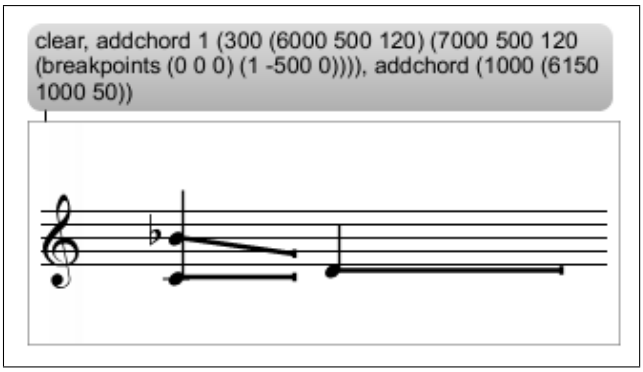


Figure 4. Any notation object in the bach library can be edited either graphically or via Max messages. In this case we clear the *bach.roll*, and then add two chords.

4.2. The GUI objects

We already introduced the *bach.score* and *bach.roll* objects. They both provide graphical interfaces for the representation of music: *bach.score* expresses time in terms of traditional musical units, and includes notions such as rests, measures, time signature and tempo; *bach.roll* expresses time in terms of absolute temporal units (namely milliseconds), and as a consequence has no notion of traditional temporal concepts: this is useful for representing non-measured music, and it also provides a simple way to deal with pitch material whose temporal information is unknown or irrelevant. It should also be noted that the implementation of traditional temporal concepts in *bach.score* is quite advanced, as it allows multiple simultaneous time signatures, tempi and agogics (see Fig. 7). Besides this fundamental difference, the two objects offer a large set of common features, including the following:

- Editing by both mouse-and-keyboard interface and Max messages (see Fig. 4): the editing facilities combine features typical of WYSIWIG music engraving tools (e.g., creation and editing of notes and chords by graphical interaction) and MIDI and audio sequencers (e.g., immediate playback of a note along with its set of meta-data - see below), and allow the user to program algorithmic, computer-driven actions upon virtually every parameter of the musical data and graphical representation. Moreover, any interaction by Max message results in the immediate update of the graphical display, and any mouse-and-keyboard interaction is properly notified: in this way, the two possible ways of interaction are seamlessly integrated with each other. Both mouse-and-keyboard and message editing operations are undoable.
- Support for microtonal accidentals of arbitrary resolution (see Fig. 5).
- Wide possibility of intervention over the graphical parameters of musical notation, including spacing (see Fig. 6), fonts and colors.

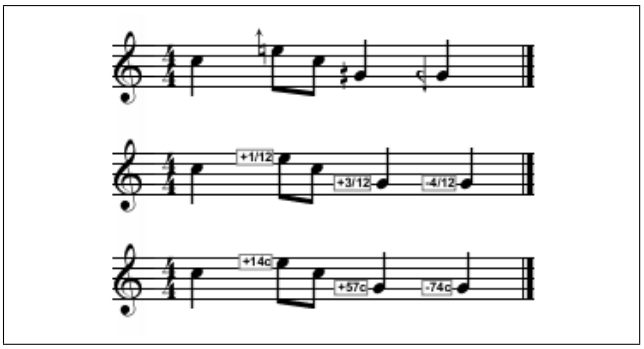


Figure 5. Semitonal, quartertonal and eighthtonal divisions are supported via the standard accidental symbols (upper example). All other microtonal division are supported as well, as long as symbolic accidentals are replaced by a label with the explicit fraction of tone (middle example), or with the cents difference from the diatonic note (lower example).

- Ability to associate to each note various types of meta-data, including text, numbers, files and breakpoint functions (see Fig. 11).
- Variable-speed playback capability: both *bach.score* and *bach.roll* can behave as advanced sequencers, and the whole set of data (such as pitch, velocity and duration information) and meta-data associated to each note is output at the appropriate time during playback, thus making both objects extremely convenient for controlling synthesizers and other physical or virtual devices.

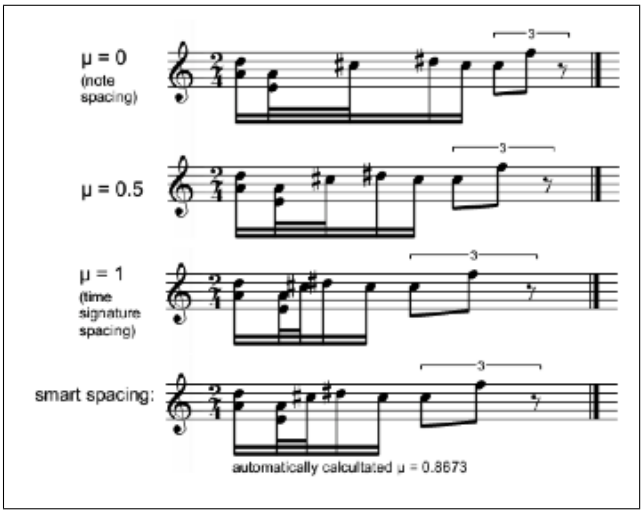


Figure 6. *bach.score* offers different spacing algorithms. The one showed here, as the parameter μ ranges in $[0, 1]$, interpolates between a note spacing ($\mu = 0$) and a time signature spacing ($\mu = 1$). The 'smart spacing' algorithm offers an automatic computation for μ (lower example).

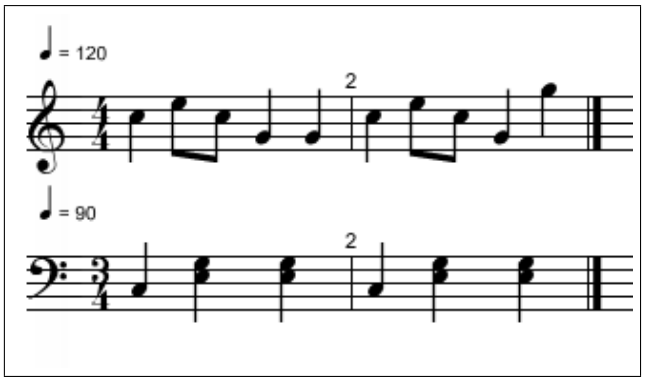


Figure 7. *bach.score* supports the cohabitation of different time signatures and tempi, and adapts the graphical alignment accordingly.

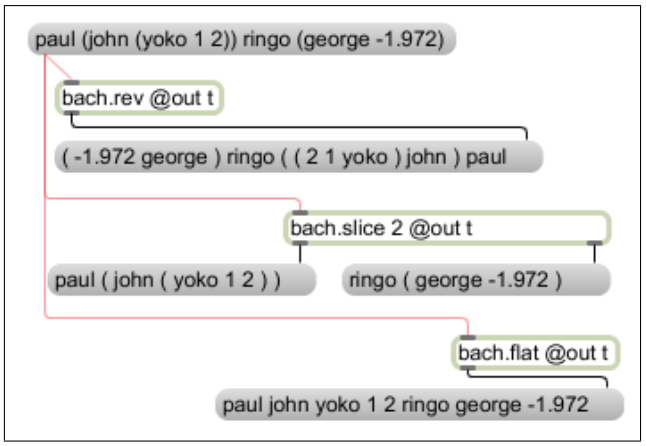


Figure 8. *bach* has a wide range of objects capable to perform standard structure operation on *llll*s, such as reversing, slicing, flattening, rotating and so on. In the picture, we see the results of reversing, slicing and flattening a list. Moreover, most operations can be constrained to some levels of depth only (e.g., one can easily remove the parentheses of all the intermediate levels, leaving the outermost and the innermost levels untouched).

4.3. Data types, representations and operators

bach also provides Max with two new data types: rational numbers and a tree structure called *llll*, an acronym for *Lisp-like linked list*. Rational numbers are extremely important in music computation, as they express traditional temporal units such as 1/2, 3/8 or 1/12 (that is, a triplet eight note) as well as harmonic ratios. The *llll* is an ordered collection of elements, each being a number (integer, floating-point or rational), a symbol (that is a text item, roughly equivalent to the concept of character string) or an *llll* - thus allowing recursively nested structures of arbitrary depth and complexity. The *llll* has been chosen for both similarity with the Lisp language, in a way to ease communication with the major existing CAC environments, and the need to establish a data structure powerful enough to represent the complexity of a musical score, but flexible enough to be a generic data container lending itself to arbitrary manipulations through a relatively small set of primitives.

In fact, the large majority of the modules of the *bach* library are tools for working upon *llll*s, performing basic operations such as retrieval of individual elements, iteration, reversal, sorting, splicing, merging and so on (see Fig. 8). Some subsets of the library are applicable to *llll*s satisfying certain given conditions: e.g., it is possible to perform mathematical operations over *llll*s solely composed by numbers; a set of operators for matrix calculus only works with appropriately structured *llll*s; and so on.

At the intersection between the modules for musical notation and the list operators is a family of objects performing operations upon *llll*s containing musical data. It is worth noting that different *bach* objects exchange musical scores in the form of specifically-structured *llll*s, whose contents is entirely readable and editable by the user; this is different from what happens e.g. in OpenMusic, where the exchange of musical data usually involves opaque objects. As a consequence, in *bach* strictly musical operations such as rhythmic quantization are just extremely specialized operations upon *llll*s, which of course can be performed only if the *llll* itself is structured as a musical score, and if its contents are consistent from the point of view of musical notation.

The structure of a *llll* representing a *bach.score* can appear quite complex at first sight, but the organization of its contents is meant to be extremely rational: after a header section containing global information such as the clefs or the types of meta-data appearing in the score, we find a sub-tree whose branches correspond to one voice each; each voice branch contains branches for each measure; each measure branch contains some measure-specific information (such as time signature) and branches for each chord; each chord branch contains some chord-specific information (such as duration) and branches for each note; and each note branch contains pitch and velocity leaves, as well as possible further specifications, such as glissando lines, enharmonic information and meta-data. The *llll* representing a whole *bach.roll* has essentially the same structure, except that the measure level is not present. With the provided set of list operators, specific pieces of information referring to single elements or sections of the score are not difficult to locate and manipulate. Moreover, both *bach.score* and *bach.roll* provide simplified ways to retrieve and enter only specific sets of values to operate upon (e.g. pitches or velocities only), which greatly ease the implementation of most algorithmic operations.

4.4. Other remarks

A choice has been made of not providing a pre-defined set of utilities performing basic musical transformations. So, *bach* does not contain specific tools for, e.g., transposition or retrogradation: but they can easily be programmed in terms of, respectively, a sum operation upon each note's pitch, or the reversal of the list containing the chords.

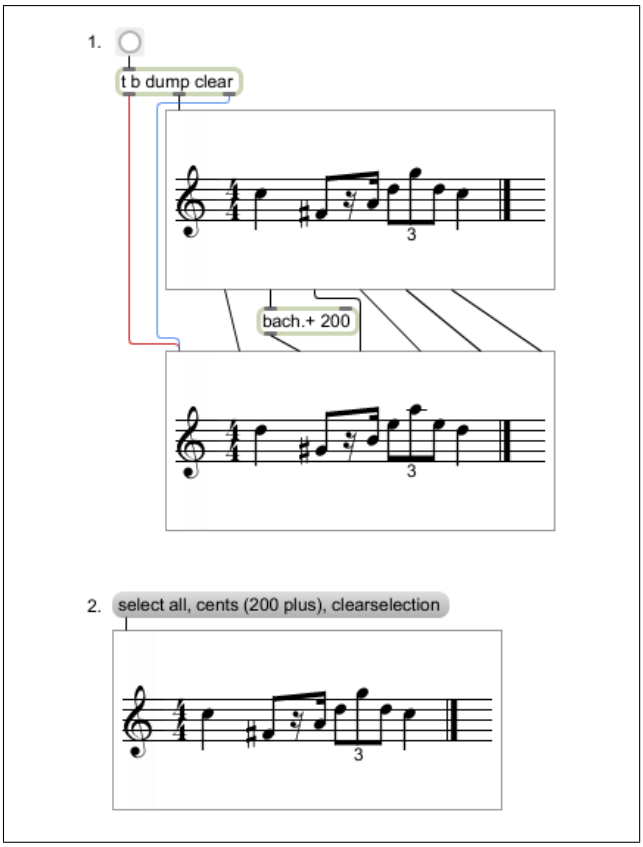


Figure 9. Two standard ways to perform a transposition in *bach*. The first one rebuilds a score having the same parameters of the first score, except for the pitches (we add a tone to each pitch). The second one achieve the transposition in-place, by selecting (by message) all notes, shifting them up, and then deselecting them (as soon as one clicks on the grey message, the result will be updated in the score).

Of course these are extremely simple cases, but the documentation provided with the library contains extensive information and examples showing how to set up several kinds of musical operations; programming *bach* should generally not be harder than, for instance, OpenMusic or PWGL, and the aim of the project is to provide a framework of tools enabling the user to program virtually any desired operation. In some future, we might develop a library of high-level tools covering a generic set of strictly musical operations; but it should be made clear that this library will be a separate project based upon *bach*, rather than a part of it. As a final note, a C API will shortly be available, allowing programmers to build new external objects for Max that will be able to access the *bach*'s data structures, low-level list operators and musical notation tools.

All this should show that *bach* is somehow placed at the intersection of several categories of musical software, as outlined in the introduction of this paper. On one hand, its capabilities of graphical representation of musical scores typically belong to music engraving systems, although it should be noted that *bach* lacks (at least in its current state) some essential features of this kind of programs, such as the ability to deal with a whole page layout, and - as a consequence - to print musical scores. On the other hand, most of its features are conceived in order to make it a tool for CAC as powerful as the traditional Lisp-based environments. Finally, it can be used as the core of an extremely advanced and flexible sequencer, with the ability to drive virtually any kind of process and playback system. And, of course, it can lend itself to innovative applications exploiting the unique convergence of these different paradigms and the real-time behavior.

5. EXAMPLES

We give here some examples of how the *bach* framework can work in practice.

5.1. A sequencer 2.0

As a first example, a score can be a customizable sequencer, whose content is completely open to any real-time process the user might want to realize.

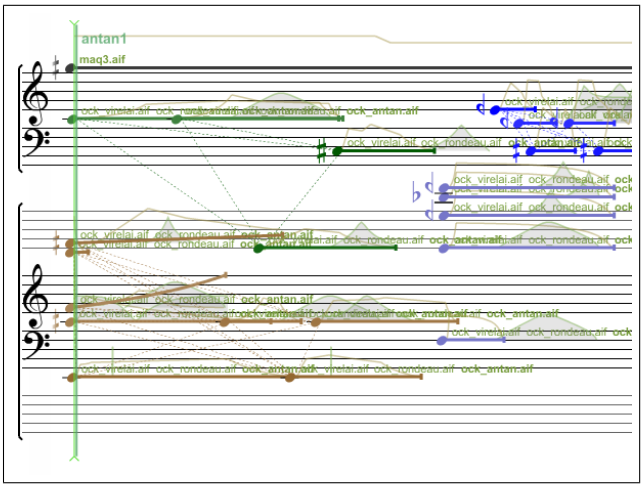


Figure 10. A portion of the electronic score for the piece *Chansons*, by D. Ghisi, consisting in a *bach.roll* object. Notes are grouped into larger structures (differently coloured), in order to better represent electronic gestures in the score.

Notes carry extra information (in some user-accessible *slots*), specifying the parameters for the processes which will concern them. This information can have various forms; to name a few: function envelopes, numeric data, filter envelopes (both static and dynamic), text instructions for a synthesizer, file paths, matrix content for routing, *llll*s, spatialization trajectories, colors. Not only all this information is accessible via messages, but it is also directly modifiable via the interface: by hitting a specific (customizable) key, one can pop out a note's slot window, and edit the information directly (see Fig. 11). The structure of each slot (data type, name, hotkey, window size, domain, range, slope...) is completely up to the user; the content of some important slots can be always kept in the

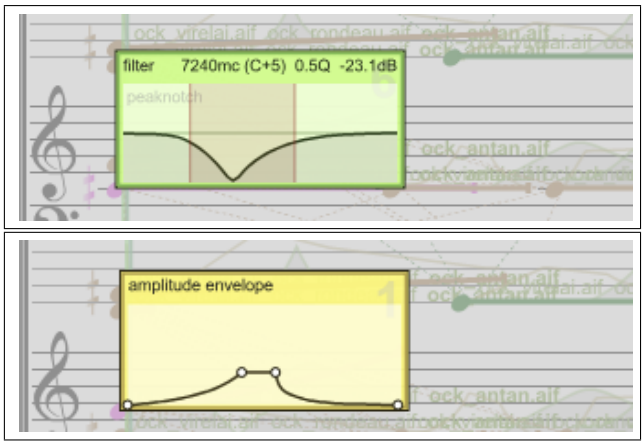


Figure 11. Two examples of slot windows.

score, so that one can constantly sees a representation of the slot content, and not only when the slot window is open (for instance, in Fig. 10 we always have under our eyes some yellow amplitude envelopes, the names of some soundfiles associated to the notes, and some filters gain responses).

With respect to other approaches to the writing of electronic music, this representation lets the user stick with traditional notation. Larger musical gestures can be handled by grouping chords (grouped chords may assume the same color and be linked with dashed lines, as one can see in Fig. 10); a group has the advantage to be shiftable as a single element, even if each note inside the group still retains its specific parameters and slot content. One can also set markers to segment portions of the score.

The slot structure is flexible enough to handle symbolic feedback. For instance, by selecting a note and pressing a key, one can link a search in a database in order to somehow match the specific pitch behavior of a note, or some data contained in its slots; one can then re-inject the result of the search in a note's slot (see Fig. 12).

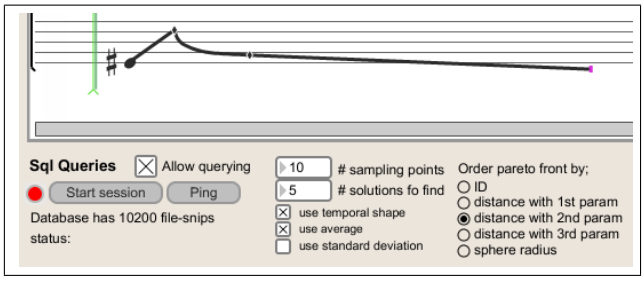


Figure 12. Portion of panel, for the piece *abroad* by D. Ghisi, querying a soundfile database, searching some closest match (with respect to the main frequency). The search engine has been developed by Philippe Esling at Ircam.

At the same time, thanks to the possibility to retrieve in real-time all the information related to the details of the graphical display of the score, it is straightforward to keep a video sequence constantly aligned to the musical score. In this way, when working with a video, one can always be aware of which video frame each musical event is synchronized to (see Fig. 13).

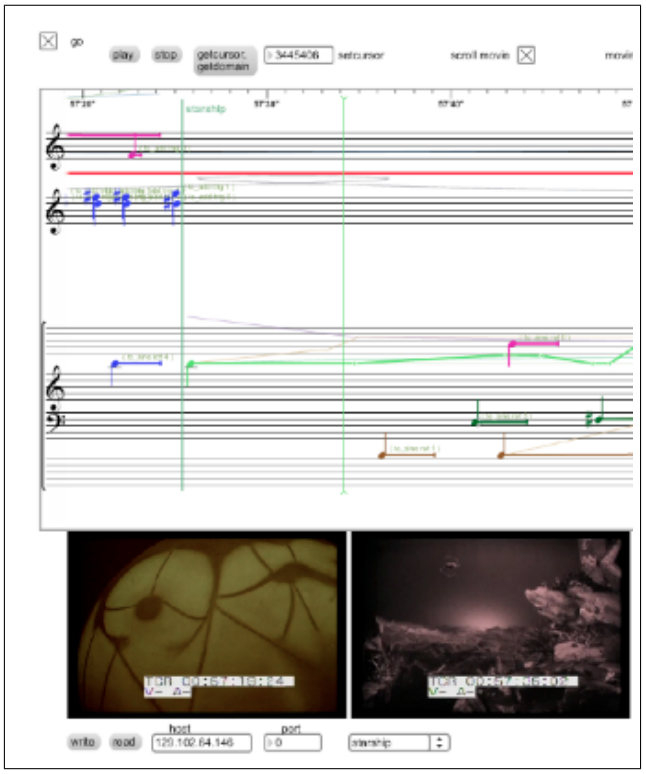


Figure 13. A portion of the patch used by A. Agostini to manage the electronic score for the film *Wunder der Schöpfung*. Each note contains a set of instructions for a synthesizer, expressed as text commands and graphical marks. When the score is played, all the information connected to each note is sent to the appropriate synthesizer voice, represented by the note color. Below the score, some frames of the movies are shown, providing a visual reference to the position of the musical events. In a separate window, not shown here, the frame corresponding to the exact position of the play cursor (the thin vertical bar) is shown in real-time, allowing for fine control over the sound and image synchronization.

5.2. A real-time symbolic granulation module

A large category of processes concerns CAC operations being performed strictly in real-time. As an example, one can easily build with *bach* a machine performing real-time granulation on symbolic data (Fig. 14). Granulation parameters can mirror the standard electroacustic granulation parameters, such as grain size, onset position, grain interval or density, grain speed, and more. When the machine is turned on, some portions of a score are extracted from a given original score (each fragment corresponding to a single grain), and then immediately integrated into a bach.roll containing the accumulated result of the granulation. A coherent playback system can be also set up, in order to have a simulated result playing accordingly. Of course, parameter changes can be handled in real-time,

depending on incoming data, such as an audio stream or a gesture tracking.

6. FUTURE DEVELOPEMENTS

At the time of writing, bach is in the phase of development called alpha in technical jargon: although the system is usable, not all the intended features have already been implemented. Namely, some important points are necessary for a complete symbolic gesture handling:

- An improved gesture hierarchy, allowing the user to give names to symbolic elements (such as chords, notes, groups), and allowing groups to be in turn grouped at higher levels. The group-wise operations need to be improved, and following this path even further, groups can have specific properties or slots.
- Support for MIDI, MusicXML and SDIF files, which constitute some of the most common ways of exchanging scores and symbolic data with other programs and devices.
- A solver for Constraint Satisfaction Problems, which are a very intuitive way to express problems according to the desired features of the solution, rather than the steps needed to build it.

Notice that the software developement situation might have changed at the time of publication, and some or all of the hereby proposed features might already be partly or fully implemented.

7. ACKNOWLEDGEMENTS

This project has been partially undertaken during the permanence of the authors at IRCAM, first as students at the Cursus in Composition and Musical Informatics and then as composers in research. We wish to thank Carlos Agon, Arshia Cont, Eric Daubresse, Emmanuel Jourdan, Serge Lemouton, Jean Lochard, Mikhail Malt and all the other people at IRCAM who have supported and encouraged the work, although not an internal IRCAM project. We wish to thank Mika Kuuskankare for his precious advice and support. We also wish to thank DaFact [6] for actively supporting and sponsoring the development of *bach*.

8. REFERENCES

[1] A. Agostini, D. Ghisi, *Real-time computer-aided composition with bach*, Contemporary Music Review, to appear.

[2] G. Assayag et al., "Computer assisted composition at Ircam: From patchwork to OpenMusic", *Computer Music Journal*, 23 (3), pages 59-72, 1999.

[3] A. Cont, *Modeling Musical Anticipation*, PhD Thesis, Paris, 2008.

[4] P. Esling, "Time Series Data Mining", *ACM Computing Surveys*, to appear.

[5] D. Fauber, Y. Orlarey, S. Letz, "INScore - An Environment for the Design of Live Music Scores", *Proceedings of the Linux Audio Conference*, LAC 2012

[6] D. Henry, "PTL, a new sequencer dedicated to graphical scores", *ICMC Proceedings*, pages 738-741, Miami, International Computer Music Association, 2004.

[7] M. Laurson, J. Duthen, "Patchwork, a graphical language in preform", *Proceedings of the International Computer Music Conference*, pages 172-175, Ann Arbor, International Computer Music Association, 1989.

[8] M. Puckette, "A divide between 'compositional' and 'performative' aspects of Pd", *Proceedings of the First Internation Pd Convention*, Graz, Austria, 2004.

[9] C. Seleborg, *Interaction temps-réel/temps différé*, mémoire ATIAM, Marseille, 2004.

[10] F. Zajéga, L. Rebousière, "Mouse Gesture Composer", *QPSR of the numediart research program*, pp. 63-70, 2010.

[6] http://www.dafact.com

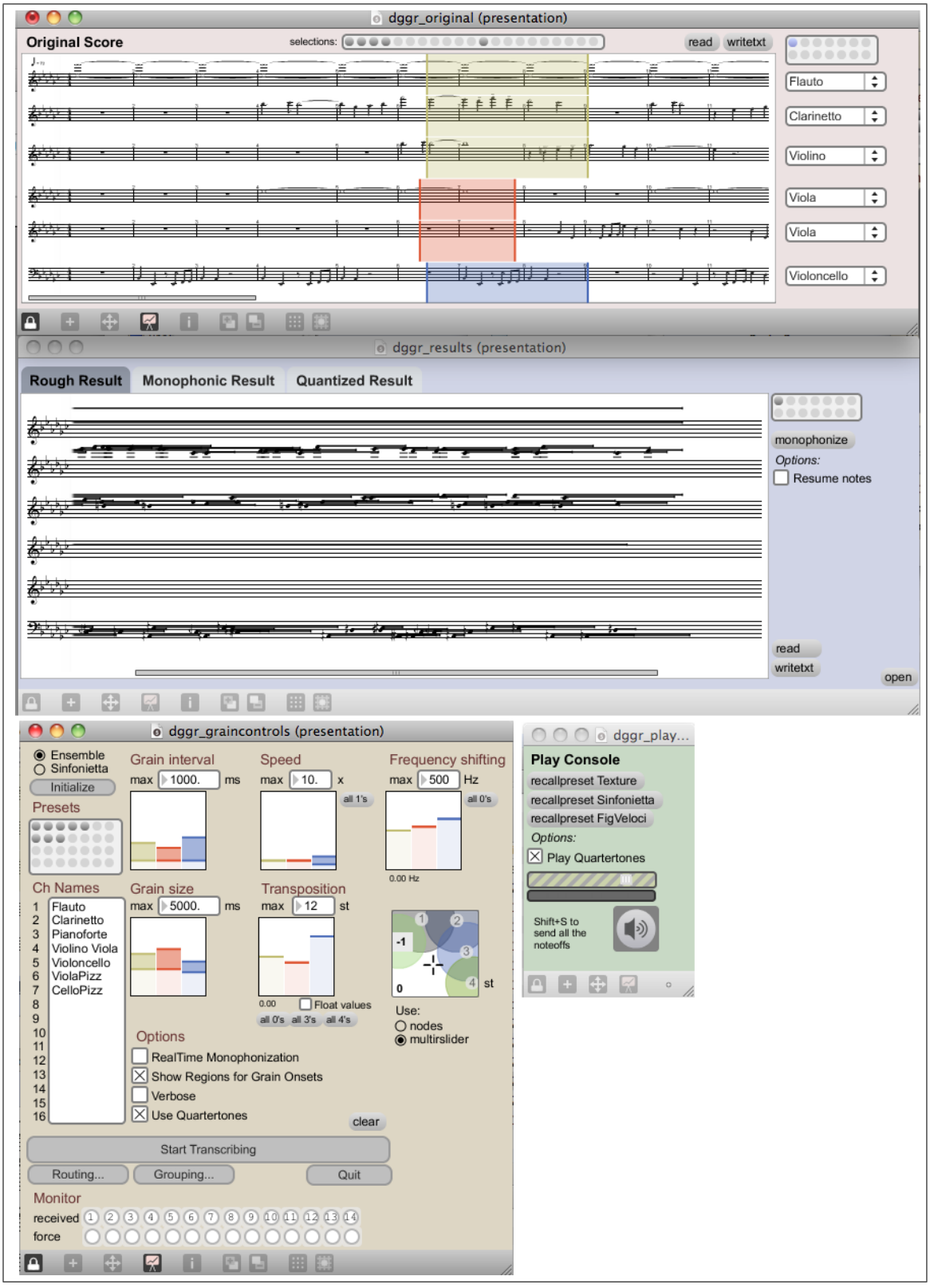


Figure 14. Screenshot of a patch achieving a real-time symbolic granulation. The original score (upper reddish window) has some markers to determine and modify the grain regions. Parameters are handled in the lower ochre window. When the user press the 'Start transcribing' button, the result appears and accumulates in the middle blue window. When desired, one may then make it monophonic (if needed), refine it, and finally quantize it. Every parameter is user-modifiable and affects in real-time the result, as in any electroacoustic granulation machine.

LA MATRICE TRIANGULAIRE, UNE GRILLE TEMPORELLE DYNAMIQUE POUR LA COMPOSITION, L'INTERPRETATION ET L'ANALYSE

Josselin Minier
IDEAT
josselin.minier@hotmail.fr

RÉSUMÉ

Comment considérer, dans un cadre musical complet, le problème du successif et du simultané ; ou plutôt, comment comprendre le simultané, en tant que phénomène émergeant du successif ?

Est envisagée une alternative à l'habituelle configuration de séquençage *matériau/temps* caractérisant l'interface de nombreux logiciels de création musicale. Le plan temporel s'apparente ici à une opposition à 90° d'un axe actuel et d'un axe virtuel, représentable par une « matrice de transfert » développant, perpendiculairement au temps linéaire, un tampon mémoriel autorisant la rétention de l'évènement passé dans une trace virtuelle actualisable en tout moment. Le matériau n'est donc pas posé *a priori*, mais se développe en tant que groupement de traces virtuelles émises par des évènements actuels. Un tel dispositif permet d'appréhender le déploiement temporel d'évènements, ce qui caractérise aussi bien l'activité de la composition que l'interprétation musicale.

La matrice de transfert peut ainsi être utilisée pour gérer des effets de filtrage temporel (granulation, délai, réverbération, *flanger*…), de transformation temporelle (*pitchshifting* et *timestretching*, ici réinterprétés sous le terme *timeshaping*), mais elle est aussi utilisée pour gérer le déploiement des effets dans le temps, et même pour effectuer une synthèse sonore dans un cadre purement temporel.

1. MANIPULATION SEQUENTIELLE

Une séquence (du latin *sequor* : suivre) implique une succession temporelle d'évènements ; selon le terme composition, l'acte de création musicale s'interprète comme une juxtaposition conjointe dans le temps d'évènements, qui prennent généralement la forme de notes. La composition pourrait alors se résumer à la génération de séquences posées les unes contre les autres, selon une structure préétablie ou qui s'établit, c'est-à-dire qui apparaît sur la table de travail. Cependant, dans nos représentations techniques, la dimension du successif est aussi développée verticalement par la simultanéité, qui accumule des strates séquentielles. Ces strates peuvent être déterminées par une caractéristique principale récurrente – un matériau à séquencer. Peu de représentations musicales échappent à ce plan matériau/temps : la partition, qui accumule les instruments ; le spectrogramme, qui accumule les fréquences ; le séquenceur multipiste, qui accumule les signaux sonores. Le séquençage peut se représenter en une matrice matériau/temps, dans laquelle sont pointés des évènements ayant un début et une fin, la fin d'un évènement ne concordant pas nécessairement avec le début du suivant. Cette configuration matricielle implique nécessairement quatre aspects temporels, dénombrables par dérivation différentielle des points de la matrice. L'évènement se situe entre deux *moments*, dont la différence correspond à l'épaisseur de *durée*. La différence entre les moments de début de deux évènements sera appelée *distance*, et la différence de durée des évènements, *écart*. Ces quatre aspects – moment, durée, distance et écart – sont primordiaux et permettent de cerner convenablement la temporalité d'un flux évènementiel, ce qui semble important en musique[1].

Si le séquençage matériau/temps, tel qu'il a précédemment été décrit, dévoile parfaitement les moments et permet de les situer les uns par rapport aux autres (avant/pendant/après), il impose toutefois l'utilisation d'une grille statique lors du repérage des autres aspects temporels ; loin d'être intuitive, cette grille doit être calculée en fonction d'une quantification métrique. Contrairement au geste instrumental, dont le mouvement entre des points spatiaux est dirigé intuitivement par l'habitude, le geste du séquençage a la nécessité du calcul de rapports temporels, parfois lourd et nuisible au mouvement créatif. Il en va de même pour le musicologue, dont les intuitions analytiques peuvent être bloquées par une imposante charge de calcul temporel et de constructions de grilles quadrillant un matériau musical défini, ou même, dans le pire des cas, prédéfini. La définition du matériau semble d'ailleurs être privilégiée au détriment de la complétude du temps par le séquençage – sélection des instruments, des notes, des sons, des effets ou des *presets*… C'est alors oublier que ce matériau musical est déjà lui-même constitué de rapports temporels de niveau inférieur (*cf.* la relation fractale *objet-structure* de Pierre Schaeffer).

[1] Dans [1], le musicologue Jean-Marc Chouvel démontre l'importance du temps musical par un algorithme de simulation cognitive de la construction de l'écoute, visuellement représentable par une matrice matériau-temps.

Une limitation du séquençage matériau-temps provient donc d'une non-explicitation des rapports temporels, réduisant la maniabilité et l'interactivité intuitive, ce qui semble mettre en cause l'ergonomie même de ce plan de travail pourtant fortement répandu. Depuis la fin des années 1990, la structuration musicale est souvent recherchée par l'intermédiaire de la matrice de similarité, telle qu'introduite par J. Foote [2], utilisée pour effectuer une segmentation [3] ou un résumé d'un morceau [4] (un état de l'art récent est proposé dans [5]). La matrice de similarité monte sur ces deux axes le même signal, en effectuant point par point un calcul grossièrement résumable à la différence entre la valeur d'un point de l'axe x et celle d'un point de l'axe y – à noter que le signal n'est généralement pas brut, mais transformé en descripteurs audio, enveloppes spectrales, MFCC, chromagrammes, rhythmogrammes, en paramètres statistiques, etc. Sans nous occuper du type de signal utilisé, la configuration de la matrice de similarité offre la possibilité de dépasser l'aspect séquentiel du signal – celui-ci étant définissable comme succession d'échantillons –, en confrontant directement différents moments. Ainsi, la valeur de signal, située sur l'axe horizontal, est confrontée à toutes les autres valeurs en chaque point de la ligne correspondante, alors qu'une valeur située sur l'axe vertical le sera en sa colonne. Un même moment du signal a donc une valeur à la fois comparée aux autres – une autre valeur lui est soustraite – et comparant les autres – elle est soustraite d'une autre valeur –, ce qui fait que les sous-triangles, divisant la matrice en deux, représentent la confrontation des mêmes valeurs, mais dans un ordre de comparaison inversé : les différences sont elles aussi inversées ; les deux triangles proposent en fait la même information, ce qui explique que la pratique n'en garde parfois qu'un seul. Cette configuration est une première étape dans la considération de tous les rapports temporels, car, si chaque moment peut être comparé aux autres, il devrait être possible de comparer cette comparaison et d'au moins déterminer les durées et les distances. Cependant, ces derniers rapports semblent difficilement repérables entre eux, puisqu'ils n'ont pas d'axe particulier auquel se référer. Les rapports temporels restent donc encore principalement circonscrits au déroulement séquentiel du temps : celui de la *timeline*, qui est ici dédoublée en deux axes.

2. UN PRINCIPE DE TRANSFERT TEMPOREL : LE TIMELAG PLUTOT QUE LA TIMELINE

L'information contenue dans une matrice de similarité se trouve principalement dans les blocs à forte valeur de similarité, ce qui indique une répétition dans la structure du signal, à un niveau local. Les diagonales de similarité, plus particulièrement, dévoilent une « non-similarité »[2] locale d'une suite de valeurs différentes qui, elle, se répète à un niveau structurel supérieur. Les durées de diagonales similaires peuvent être évaluées comparativement, puisque ces diagonales se situent sur une même ligne ou une même colonne de la matrice. Les durées des diagonales non-similaires, ne partageant pas de coordonnées, semblent difficilement comparables en termes de durées. C'est peut-être là un travers de cette représentation, qui favorise la découverte de répétitions linéaires d'éléments structurels, tout en écartant les variations.

La confrontation de durées et de distances nécessite un axe dédié non pas au moment, comme c'est le cas dans une *timeline*, mais à la différence entre moments, c'est-à-dire la durée. Une autre forme de matrice développe un axe vertical que l'on pourrait qualifier de *timelag*, puisqu'élaboré par un décalage temporel. Ce n'est plus le moment qui est représenté, mais bien la durée de décalage. La matrice de forme triangulaire se construit en fait selon un principe cognitif de rétention mémorielle de l'information, qui n'est pas sans rappeler les diagrammes temporels d'Husserl[3], ou la construction de l'épaisseur du présent dans la synthèse temporelle de Bergson, ensuite reprise par Deleuze[4]. L'axe horizontal représente une *timeline* ordinaire, où chaque moment est prolongé dans les moments suivants, tout en s'en différenciant sur l'axe vertical. Chaque moment de la *timeline* est conservé au niveau du moment suivant en se décalant sur le *timelag*, selon un transfert continuel formant une diagonale, comparable à une trace mémorielle conservant le moment passé. Au niveau cognitif, le transfert est susceptible d'entraîner une dégradation de la trace, ce qui constitue, nous le développerons ensuite dans le cadre des effets sonores, une configuration intéressante et dynamique de la matrice triangulaire.

Calcul d'un point de la matrice à partir du *timelag* :

$$M(t,d) = m_{t-d}$$

M étant la matrice sur un plan temps/durée (*T*/*D*), et *m* le moment qu'un point de la matrice représente. Une ligne verticale représente les moments actuels *t*, une ligne horizontale représente les durées de décalages *d*, et une ligne diagonale représente les moments virtuels $t-d$, c'est-à-dire qu'en un moment actuel en *t* est transféré un autre moment passé selon un décalage *d*. Puisque l'axe *D* est celui des durées, l'espace, dans la matrice, entre deux moments actuels est égal à la durée en *D*, formant ainsi un triangle rectangle en son angle bas droit – représentant le moment actuel de perspective –, dont l'hypoténuse relie l'angle bas gauche

[2] Plutôt que de similarité, il serait plus cohérent de parler de non-différence ou même de faible différence, puisque le signal sonore se répète rarement de manière parfaite, d'autant qu'une différence essentielle intervient entre deux segments « similaires » : leur localité. Donc « similaire » ne signifie pas « identique », et encore moins « même ». Pour ces considérations plus philosophiques, nous renvoyons à Gilles Deleuze.

[3] Remercions Gérard Assayag et Jean-Marc Chouvel de nous avoir présenté les similitudes entre la matrice triangulaire et les diagrammes husserliens.

[4] L'aspect cognitif et philosophique de la construction de la matrice triangulaire est développée dans [6] et [7].

– représentant le moment actuel inspecté –, et l'angle du haut – le moment virtuel, point de rencontre entre les deux moments actuels –. Autrement dit, le moment de perspective correspond à un *présent*, le moment inspecté, à un *passé*, et le moment virtuel peut donc être considéré comme un *passé dans le présent*.

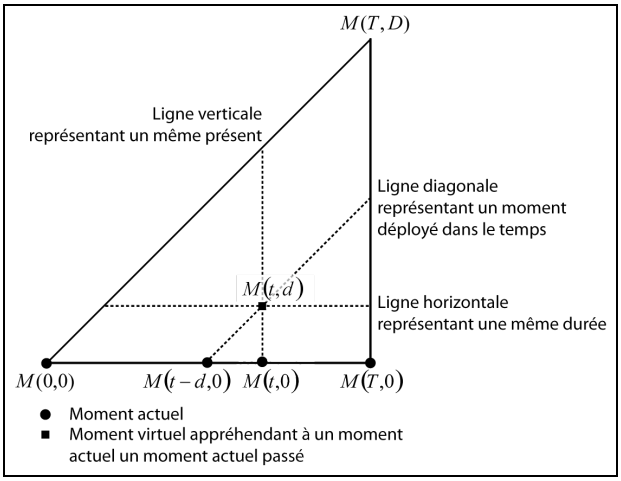


Figure 1. Plan d'une matrice triangulaire.

Dans la figure 1, une matrice triangulaire est construite sur un rapport temporel simple. Sur la *timeline* – la ligne horizontale du bas de la matrice, en $M(T)$, représentant l'actualité du temps –, sont disposés des moments se succédant conjointement. Un moment actuel inscrit dans la matrice en $M(t,0)$ peut appréhender un moment $M(t-d,0)$ qui lui est passé de la durée *d*; le rapport entre ces deux moments étant inscrit dans la matrice en $M(t,d)$, une diagonale relie ce point au point $M(t-d,0)$, représentant ainsi le parcours dans le temps d'un moment $t-d$ jusqu'à t. De chaque moment actuel se déploie donc une diagonale le représentant virtuellement dans son futur. Cette diagonale croise des droites verticales reliant $M(t,0)$ à $M(t,d)$, et représentant des moments virtuels compris à un moment actuel ; à l'intersection d'une diagonale et d'une droite verticale peut être effectuée une opération sur des valeurs des deux moments pointés, rapprochant ainsi la matrice triangulaire d'une matrice de similarité. Mais cette intersection a aussi une valeur hors de celle des moments, puisqu'elle détermine le *lag*, le décalage, c'est-à-dire la durée de l'évènement encadré. De la même manière, peut être déterminée la distance entre évènements, ainsi que la distance réduite, comprise entre un moment de fin d'un évènement et un moment de début d'un autre (figure 2). Ces rapports temporels peuvent être examinés facilement grâce à l'axe du *timelag* qui permet de classer les durées en fonction de leur hauteur. L'écart de durée de deux évènements est alors déterminé par l'espace compris entre une intersection représentant une durée, et une droite horizontale passant par une intersection représentant une durée inférieure.

Des sous-triangles sont tracés dans la matrice de la figure 2, pour représenter une similarité de l'espace temporel. Une application informatique pourrait permettre de transférer des triangles pour comparer ou modifier la temporalité de la séquence, en les faisant simplement glisser par geste. D'ailleurs, puisque les sous-triangles composant la matrice sont nécessairement rectangles isocèles, ils peuvent être construits ou modifiés en calibrant simplement l'étendue de la diagonale formant leur hypoténuse, ce qui autorise une manipulation rapide. Cette manipulation peut alors être qualifiée de vectorielle, et opposée en cela à la manipulation purement matricielle du séquençage traditionnel – cette opposition fait directement référence à celle entre le dessin matriciel manipulant des pixels, et le dessin vectoriel manipulant des vecteurs.

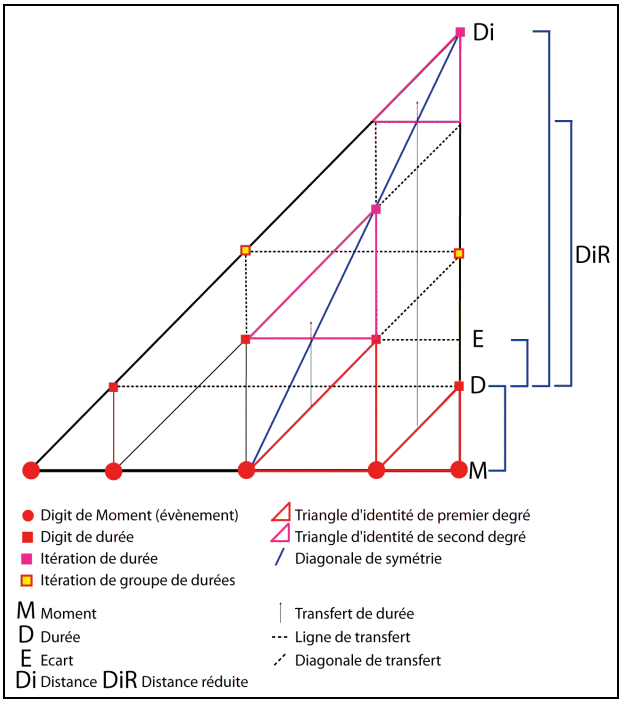


Figure 2. Matrice triangulaire – exemple d'analyse d'une séquence temporelle.

L'aspect vectoriel comporte un avantage, qui est celui de la scalabilité. Les vecteurs permettent de sélectionner ou de conserver différents rapports temporels sur différentes échelles. Les tables d'ondes[5] utilisées pour la granulation du signal, ainsi que les séquenceurs de type Cubase et ProTools, posent un problème de manipulation de rapports temporels et de scalabilité. Dans la version élémentaire d'un séquenceur, la modification (manuelle) uniforme des rapports temporels d'une séquence, formée de blocs de signal, impose une modification proportionnelle de la grille de tempo ; seule la distance entre les blocs doit être modifiée, indépendamment de leur durée ; les blocs nouvellement espacés doivent ensuite être conservés

[5] Une table d'onde peut être appréhendée en tant que fragment de signal séquencé sur une *timeline*.

sans aucune modification, alors que la grille de tempo revient quant à elle à sa valeur d'origine. Une telle manipulation (par exemple le *Time Warp* dans Cubase) est couteuse et parfois rébarbative, alors que la modification est uniforme. Dans une matrice triangulaire, elle pourrait être simplement effectuée par un glisser-écarter agrandissant les triangles d'une matrice triangulaire. Bien plus encore, dans un séquenceur traditionnel, une interpolation entre deux séquences n'est pas directement effectuable manuellement, puisque les rapports temporels ne sont pas explicites. L'interpolation de séquences, permettant de passer de l'une à l'autre de manière progressive, peut s'opérer en reliant les triangles et en les transférant graduellement, c'est-à-dire selon une graduation déterminée par une sous-grille à pas constant ou variable, la variation du pas étant elle-même précisément définissable par vecteur.

3. MANIPULATION VECTORIELLE

La manipulation vectorielle peut s'exprimer plus conventionnellement grâce à quelques formules comparables à celles utilisées en traitement du signal. Sur une matrice *T/D*, l'opération de base, calculant la différence entre des points de la matrice, peut s'écrire :

$$M_{diff}(t,d) = m_t - m_{t-d}$$

où m_t est le point actuel et m_{t-d} le point virtuel transféré. Un effet de délai peut alors en être simplement déduit pour chaque point de la matrice :

$$M_{del}(t,d) = m_t + m_{t-d}$$

où *d* correspond à la durée du délai. Il est possible de pondérer les différents moments par un coefficient $\alpha = [0;1]$:

$$M_{del}(t,d) = (1-\alpha)m_{t-d} + \alpha m_t$$

On retrouve bien là la structure des filtres RIF (Réponse Impulsionnelle Finie) servant à l'élaboration des délais simples. Une réponse impulsionnelle infinie (RII), permettant la construction de délais récursifs, peut être obtenue en mettant en rapport un moment du signal brut et un point précédant de matrice, c'est-à-dire en intégrant par réinjection la sortie du précédant calcul :

$$M_{del}(t,d) = (1-\alpha)M_{del}(t-d,d) + \alpha m_t$$

où $M_{del}(t-d,d)$ correspond à un point de la matrice précédemment calculé.

Une utilisation en temps réel de la matrice triangulaire[6], développée dans le logiciel Pure Data, permet de manipuler un fichier sonore numérique, en contrôlant la temporalité de lecture par le biais de vecteurs. Les configurations vectorielles se rapprochent ainsi de divers effets traditionnellement utilisés en traitement du signal[7], bien qu'elles n'en fournissent qu'une version simplifiée, mais ergonomiquement intéressante et propice à l'imagination.

Un triangle de lecture peut ainsi glisser sur le vecteur (droite rouge, en figure 3), tel que celui tracé sur la matrice 1 de la figure 3, pour construire un délai simple, dont la durée est déterminée par la position de la ligne en fonction du *timelag*. À chaque moment de la ligne, sont lus simultanément l'échantillon du signal situé au même moment, ainsi que l'échantillon passé, pointé par la diagonale du triangle.

Une simple interpolation permet de calculer le vecteur :

$$V(t,d) = (t - deb_T)\frac{fin_D - deb_D}{fin_T - deb_T}$$

$fin_D - deb_D$ correspond à l'espace en *D* du vecteur, $fin_T - deb_T$ à son espace en *T*, et $(t - deb_T)$ à la situation du moment *t* du calcul dans ce dernier espace. Puisque la hauteur en *D* renvoie à un moment passé, celui correspondant à un point du vecteur peut simplement être trouvé en soustrayant la valeur *d* du moment *t* du vecteur :

$$V(t,d) = (t - deb_T)\frac{fin_D - deb_D}{fin_T - deb_T} - d$$

Une manipulation en temps réel de la hauteur de la ligne permet de modifier la durée du délai, alors qu'une inclinaison de la ligne modifie le pas de lecture des échantillons, et donc la vitesse. En effet, comme le montre la matrice 2, plus la pente est négative, plus la vitesse est élevée, et inversement, ce qui s'explique par le fait que la totalité du segment original est lue sur une durée plus réduite. Comme le montre la matrice 3, plusieurs vecteurs peuvent être ajoutés, permettant de réitérer plusieurs fois un évènement, comme le contrôlerait le *feedback* d'un délai à réinjection. Cependant, contrairement au délai, la matrice permet de manipuler indépendamment chaque vecteur – et donc chaque itération –, que ce soit en durée ou en vitesse. La manipulation d'un sous-vecteur peut aussi modifier le rapport entre un ensemble de vecteurs (matrice 4).

[6] Voici quelques liens vers des vidéos de démonstrations et d'improvisations par matrice triangulaire en configuration *live* simplifiée : www.dailymotion.com/Josselin_Minier

[7] Des effets numériques sont décrits dans une thèse de traitement du signal s'intéressant principalement à leur aspect adaptatif [8].

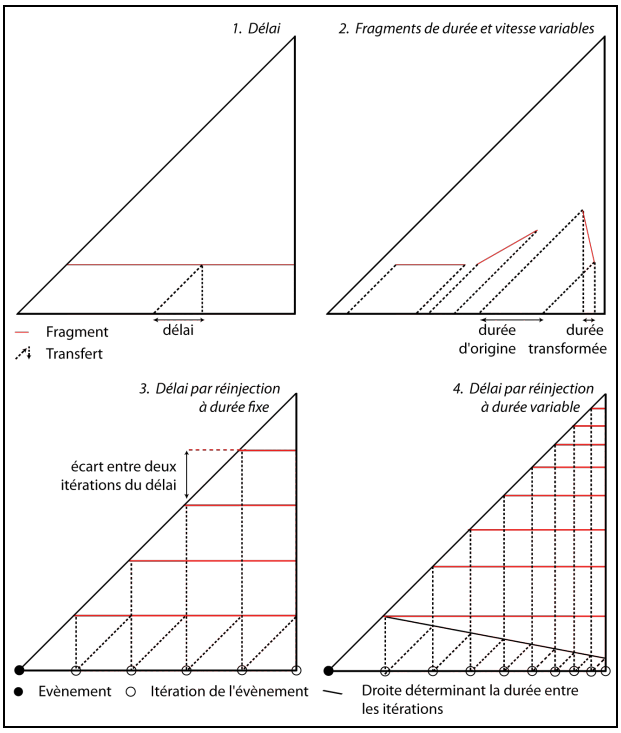


Figure 3. Matrices triangulaires – lecture d'une table (*wavetable*, table d'effets).

Si une lecture « à deux têtes » est possible, l'une correspondant au moment de la diagonale du triangle et l'autre au moment de la droite verticale, nous pouvons aussi bien choisir de n'utiliser qu'une seule « tête de lecture ». Par exemple, une réitération d'un segment s'effectue avec une seule diagonale, dirigée par une figure en escalier à marches de durées constantes (matrice 5). Cette durée de réitération peut être modulée par un sous-vecteur contrôlant les bords des marches (matrice 6). Si aucun de ces sous-vecteurs de bord de marche ne correspond à une diagonale, la réitération se décalera au fur-et-à-mesure, aboutissant ainsi à un effet similaire à un *timestretching*, qui ne s'occuperait pas encore des problèmes de pondération du fenêtrage des segments et ne respecterait pas la phase. Utilisons le terme *timeshaping*, puisque cela introduit une distorsion du signal, tout comme le fait le *waveshaping*, dans le domaine de l'intensité. Peuvent cependant être aisément ajoutées des fonctions de chevauchement (comparable à l'*overlapping*) et de pondération de fenêtrage. Enfin, complétons les possibilités offertes par la matrice triangulaire avec une fonction d'interpolation entre fragments déterminés par différents vecteurs.

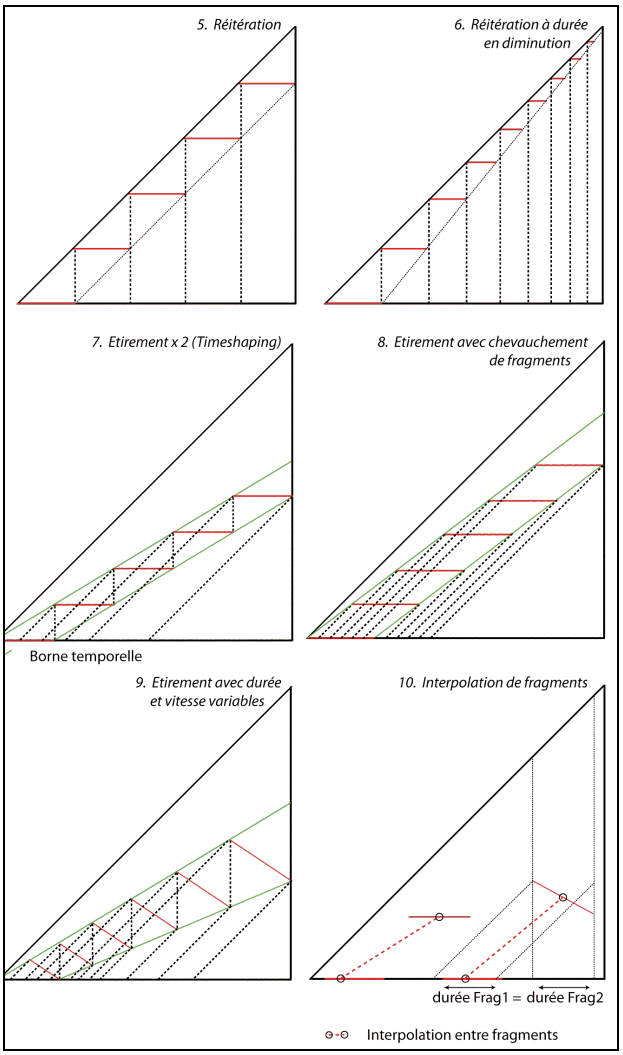


Figure 4. Matrices triangulaires – 2.

4. CONCLUSION

Se posant en alternative du séquençage par matrice matériau-temps n'explicitant pas pleinement les rapports temporels et limitant en cela les manipulations temporelles envisageables, la matrice triangulaire permet de concevoir une construction musicale plus proche du temps – celui développé par Bergson, Husserl et Deleuze –. Diverses figures temporelles s'approchant des filtrages utilisés en traitement du signal sont rapidement traçables à la manière du dessin vectoriel. Cette utilisation de vecteurs orientant la lecture de fragments sonores offre de multiples possibilités de manipulation et permet une représentation efficace des méthodes granulaires ou à table d'ondes ; et, dans un rôle similaire à celui des procédures de classements d'éléments sonores dans les méthodes par concaténation, une contrainte peut prédéfinir l'allure des vecteurs. Un exemple de contrainte peut être donné, sans développer outre mesure : une première analyse multi-échelles du signal permet de générer une matrice de fragmentation (*cf.* figure 5) dans laquelle chaque ligne horizontale regroupe des enveloppes aux bornes représentant les début et fin des fragments du signal. Un

vecteur peut donc suivre ces lignes de la matrice, ou suivre un parcours de fragments à définir, les *clics* parasites de transition étant éliminés en multipliant le fragment de signal par l'enveloppe correspondante.

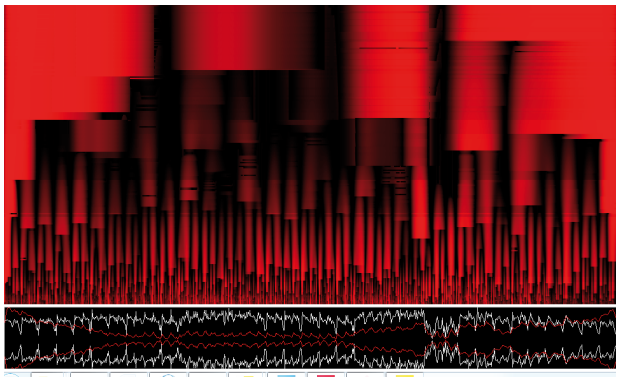


Figure 5. Matrice de fragmentation du signal, aussi appelée « matrice en orgue »[8], représentant des enveloppes de fragments (« Cocoon », *Vespertine*, Björk, 2001) ; des vecteurs de matrice triangulaire peuvent suivre une ligne ou un parcours de fragments.

Enfin, notons que les vecteurs semblent d'autant plus intéressants qu'ils sont indépendants de la nature des données. Si l'utilisation proposée dans cet article se concentre sur le signal sonore numérique, les données manipulables sous cette configuration triangulaire peuvent aisément s'étendre aux courbes d'automation paramétrant les effets utilisés dans les logiciels de création sonore, de composition et d'interprétation musicale. Les capacités de réalisation intuitive de calculs sur le temps, par tracé de vecteur et transfert de triangle, autorisent aussi une utilisation musicologique visant à la compréhension de la construction temporelle de la musique.

5. REFERENCES

[1] Chouvel, J-M. *Analyse musicale. Sémiologie et cognition des formes temporelles*. L'Harmattan, Paris, 2006, 314 p.

[2] Foote, J. " Visualizing Music and Audio using Self-Similarity", Proceedings of ACM Multimedia, Orlando, USA, 1999, pp. 77-80.

[3] Jensen, K. "Multiple Scale Music Segmentation Using Rhythm, Timbre, and Harmony", EURASIP Journal on Advances in Signal Processing, Heidelberg, Germany, 2007, 11 p.

[4] La Burthe, A. ; Peeters, G. ; Rodet, X. "Toward Automatic Music Audio Summary Generation from Signal Analysis", Proceedings of the 3rd International Society for Music Information Retrieval Conference, Paris, France, 2002, 7 p.

[5] Klapuri, A. ; Müller, M. ; Paulus, J. " Audio-based music structure analysis", Proceedings of the 11th International Society for Music Information Retrieval Conference, Utrecht, Netherlands, 2010, 12 p.

[6] Minier, J. *Fragmentation cognitive/informatique de la musique populaire amplifiée. Construction d'un système dirigé par une notion de simulacre cinétique*. Thèse, Paris I Sorbonne Panthéon, 2011, 368 p.

[7] Minier, J. *Synthèse du temps et fragmentation cognitive du sonore*, Séminaire « Philosophie et Musicologie, des croisements aux rencontres », 2012, actes à paraître, 9 p.

[8] Verfaille, V. *Effets audionumériques adaptatifs : théorie, mise en œuvre et usage en création musicale numérique.* Thèse, ATIAM, Aix-Marseille II, 2003, 326 p.

[8] La procédure de la matrice en orgue est expliquée dans [6].

COMPOSITION DE PARTITIONS MUSICALES

D. Fober, Y. Orlarey, S. Letz
Grame - Centre national de création musicale
{fober, orlarey, letz}@grame.fr

RÉSUMÉ

Basé sur le format de notation musicale GUIDO, nous avons développé un ensemble d'outils pour la composition de partitions musicales. Il s'agit d'opérations de transformation de haut niveau, prenant des partitions en entrée pour en générer de nouvelles en sortie. Ces opérations peuvent par exemple s'appliquer au domaine temporel (e.g. couper le début ou la fin d'une partition) ou concerner la structure de la partition (mise en séquence, en parallèle). La définition d'opérateurs applicables au niveau de la notation permet d'englober l'expression d'idées musicales et leur composition dans une même métaphore. Cela soulève toutefois un certain nombres de problèmes liés à la cohérence de la notation. Cet article donne un aperçu du format de notation musicale GUIDO, puis il présente les opérations de composition de partitions, les problèmes qui se posent pour la cohérence de la notation et les solutions proposées.

1. INTRODUCTION

Le format de notation musicale GUIDO [GMN] [1] [2] a été défini par H. Hoos et K. Hamel il y a plus de 10 ans. Il est très proche du format adopté par Lilypond [3] [4] mais il est apparu antérieurement. Le format GMN est un langage textuel de représentation de partitions musicales. Il est basé sur un formalisme simple mais puissant, se concentrant sur les grands concepts musicaux (en opposition aux caractéristiques graphiques). L'approche développée par GUIDO repose sur l'idée *d'adéquation*, qui invite à noter simplement les concepts musicaux simples, et à ne recourir à des notations complexes que pour les notions musicales complexes.

Basée sur le format GMN, la librairie GUIDO [5, 6] fournit un puissant moteur de mise en page de partitions, qui se différencie notamment des approches de type *compilateur* [3, 7] par sa capacité à être embarquée dans une application indépendante, et par la rapidité et l'efficacité du moteur de rendu, qui le rendent utilisable dans un contexte *temps réel* pour des partitions simples.

Basées sur la combinaison du langage et du moteur de rendu GUIDO, des opérations de composition de partitions ont été développées : opérations de transformation rythmique, de hauteur, de sélection temporelle ou événementielle, de mise en séquence ou en parallèle, etc.

La notation musicale reste l'une des représentations les plus utilisées par les musiciens. La définition d'opérateurs de composition de partitions constitue un moyen homogène de concilier écriture et transformation tout en restant à un haut niveau de description symbolique de la musique. De plus, la conception de ces opérateurs permet d'utiliser des partitions tant comme cible que comme argument de ces opérations, renforçant la métaphore de la notation comme support d'idées musicales aussi bien que d'opérations de composition.

Toutefois, ces opérations de composition de partitions musicales soulèvent un certain nombre de problèmes liés à la cohérence de la notation. Comme élément de réponse à ces problèmes, nous proposons une typologie simple des éléments de notation musicale ainsi qu'un ensemble de règles de composition basées sur cette typologie.

Cet article donnera quelques rappels sur le format de notation GUIDO, les sections suivantes présenteront les opérations de composition de partitions, les problèmes soulevés par ces opérations et les solutions proposées. Nous proposerons ensuite une extension du langage GMN pour prendre en compte la réversibilité des opérations.

2. LE FORMAT DE NOTATION GUIDO

2.1. Concepts de base

Le format de base de GUIDO recouvre les notes, silences, altérations, voix indépendantes, et les concepts les plus courants de la notation musicale comme les clefs, métriques, armures, articulations, liaisons, etc. Les notes sont représentées par leur nom `(a b c d e f g h)`, une altération optionnelle ('#' et '&' pour dièse et bémol), un numéro d'octave optionnel et une durée optionnelle.
La durée est spécifiée par l'une des formes suivantes :

```
'*'enum'/'denom dotting
'*'enum dotting
'/'denom dotting
```

où $enum$ et $denom$ sont des entiers positifs et $dotting$ est soit vide, '.', ou '..', avec la même sémantique que dans la notation musicale. 1 est la valeur par défaut si $enum$ ou $denom$ sont omis. La durée exprime une fraction de ronde.

Si omises, les valeurs optionnelles sont les dernières utilisée pour la note précédente de la séquence.

Les accords sont décrits par des notes entre accolades séparées par des virgules, i.e. `{c, e, g}`

2.2. Tags GUIDO

Les tags sont utilisés pour donner des informations musicales supplémentaires, comme les liaisons, clés, ar-

mures, etc. Un tag a l'une des formes suivantes :

```
\tagname
\tagname<param-list>
```

où `param-list` est une liste de chaînes de caractères ou de nombres, séparés par des virgules. Un tag peut avoir une étendue temporelle et s'appliquer à un ensemble de notes (par ex. les liaisons) ; la forme correspondante est :

```
\tagname(note-series)
\tagname<param-list>(note-series)
```

Le code GMN suivant illustre la concision de la notation ; la figure 1 donne le rendu effectué par le moteur GUIDO.

```
[ \meter<"4/4"> \key<-2> c d e& f/8 g ]
```

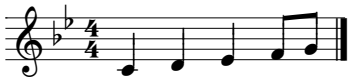


Figure 1. Un exemple GMN simple

2.3. Séquences et segments

Une séquence de notes a la forme `[tagged-notes]` où `tagged-notes` est une série de notes, tags, tags temporels séparés par des espaces. Une séquence de notes représente une seule voix. Les segments de notes représentent plusieurs voix ; ils sont notés par `{seq-list}` où `seq-list` est une liste de séquences de notes séparées par des virgules, comme dans l'exemple suivant (figure 2) :

```
{ [ e g f ], [ a e a ] }
```

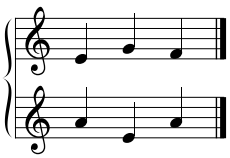


Figure 2. Exemple à plusieurs voix

2.4. GUIDO Avancé

Le format GUIDO Avancé inclus des tags supplémentaires et un contrôle plus fin de la mise en page. Il définit notamment des paramètres tels que dx et dy pour ajuster le positionnement des éléments de la partition. Il autorise le formatage de notes et de silences, les assignations à des portées, etc. Un exemple de code est donné ci-dessous avec le rendu correspondant (figure 3).

```
{
 [
  \barFormat<"system">
  \staff<1> \stemsUp \meter<"2/4">
  \intens<"p", dx=1hs,dy=-7hs>
  \beam(g2/32 e/16 c*3/32) c/8
  \beam(\noteFormat<dx=-0.9hs>(a1/16) c2 f)
  \beam(g/32 d/16 h1*3/32) d2/8
  \beam(h1/16 d2 g)],
  [\staff<1>\stemsDown g1/8 e
   f/16 \noteFormat<dx=0.8hs>(g) f a a/8 e
   f/16 g f e],
  [\staff<2> \meter<"2/4">
   \stemsUp a0 f h c1],
  [\staff<2> \stemsDown c0 d g {d, a}]
}
```

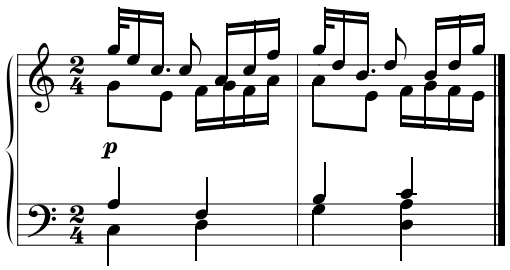


Figure 3. Exemple GUIDO Avancé

3. COMPOSITION DE PARTITIONS

3.1. Operations

Des opérations de composition de partitions ont été implémentées sous forme de librairie indépendante (GuidoAR). Elles sont décrites par la table 1. Ces opérations sont disponibles sous forme d'API, d'utilitaires pour la ligne de commande, ou encore dans l'environnement graphique d'une application indépendants : GUIDOCalculus. La plupart de ces opérations prennent une partition et une valeur en entrée pour produire une nouvelle partition en sortie. La valeur passée en entrée peut-être donnée sous forme de partition : par exemple, l'opération `top` coupe les voix d'une partition après un nombre de voix n ; en utilisant une partition à la place de ce paramètre, c'est le nombre de voix de cette partition qui sera pris comme valeur. Ainsi, toutes les opérations peuvent se décrire de manière homogène au niveau symbolique de la notation.

Ce design permet également d'exprimer des séries de transformations en ligne de commande, comme le pipeline de partitions à travers de opérateurs e.g.

```
head s1 s2 | par s2 | transpose "[ c ]"
```

3.2. Les problèmes de notation

Dans les faits, les fonctions de composition de partition opèrent sur une représentation en mémoire de la notation musicale. Toutefois, nous allons illustrer les problèmes liés avec la représentation textuelle qui est équivalente à la représentation mémoire.

Prenons l'exemple de l'opération `tail` appliquée à la partition suivante :

```
[\clef<"f"> c d e c]
```

Une coupure *brute* de la partition après 2 notes donne le code `[e c]` où l'information de clef a disparu, pouvant conduire à un rendu non souhaité (figure 4).

Voici un autre exemple avec l'opération `seq` : la mise en séquence de `[\clef<"g"> c d]`

operation	args	description
seq	$s1$ $s2$	met $s1$ et $s2$ en séquence
par	$s1$ $s2$	met $s1$ et $s2$ en parallèle
rpar	$s1$ $s2$	met $s1$ et $s2$ en parallèle, alignés à droite
top	$s1$ $[n \mid s2]$	prend les n premières voix de $s1$; quand on utilise une partition $s2$ comme paramètre, n est le nombre de voix de $s2$
bottom	$s1$ $[n \mid s2]$	coupe les n première voix de $s1$; quand on utilise une partition $s2$ comme paramètre, n est le nombre de voix de $s2$
head	$s1$ $[d \mid s2]$	prend le début de $s1$ jusqu'à la date d ; quand on utilise une partition $s2$ comme paramètre, d est la durée de $s2$
evhead	$s1$ $[n \mid s2]$	prend les n premiers événements $s1$; quand on utilise une partition $s2$ comme paramètre, n est le nombre d'événements de $s2$
tail	$s1$ $[d \mid s2]$	coupe le début de $s1$ jusqu'à la date d ; quand on utilise une partition $s2$ comme paramètre, d est la durée de $s2$
evtail	$s1$ $[n \mid s2]$	coupe les n premiers événements de $s1$; quand on utilise une partition $s2$ comme paramètre, n est le nombre d'événements de $s2$
transpose	$s1$ $[i \mid s2]$	transpose $s1$ d'un intervale i ; quand on utilise une partition $s2$ comme paramètre, i est calculé comme la différence entre la première note de la première voix de $s1$ et $s2$
duration	$s1$ $[d \mid r \mid s2]$	étire $s1$ à une durée d ou selon un facteur r ; quand on utilise une partition $s2$ comme paramètre, d est la durée de $s2$
applypitch	$s1$ $s2$	applique les hauteurs de $s1$ à $s2$ en boucle
applyrythm	$s1$ $s2$	applique le rythme de $s1$ à $s2$ en boucle

Table 1. Opérations sur les partitions

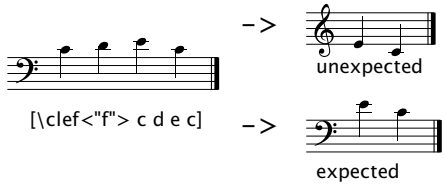


Figure 4. Cohérence de l'opération `tail`

et `[\clef<"g"> e c]`
donnera `[\clef<"g"> c d \clef<"g"> e c ]`
où la clef est répétée inutilement (figure 5) ce qui rend la lecture plus confuse.

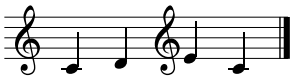


Figure 5. Mise en séquence *brute*

Certaines opérations peuvent également donner des résultats syntactiquement incorrects. Considérons le code suivant :

`[g \slur(f e) c]`

le partage de la partition en 2 après `f` donnerai

a) `[g \slur(f]` et **b)** `[e) c]`

i.e. comprenant des tags temporels inachevés. Nous utiliserons les termes tags à *fin-ouverte* pour faire référence au cas a) et tag à *début-ouvert* pour le cas b).

Ces quelques exemples illustrent une partie du problème et il y a beaucoup d'autres cas où la cohérence de la notation doit être préservée lors d'opérations sur les partitions.

3.3. Etendue temporelle des éléments

Nous proposons de traiter le problème en définissant une typologie des éléments de notation en fonction de leur étendue temporelle, et en spécifiant des règles de cohérence basées sur ces types.

Le format GMN fait une distinction entre les tags de position (comme `\clef` ou `\meter`) et les tags temporels (`\slur`, `\beam`, etc.) :

- les tags temporels ont une durée explicite : la durée des notes qu'ils contiennent,
- les tags de position sont de simples marques de notations à une position donnée.

Cette distinction n'est toutefois pas suffisante pour couvrir le problème : beaucoup de tags de position ont une durée implicite, qui s'étend généralement jusqu'à un signe de de notation similaire ou jusqu'à la fin de la partition. C'est le cas par exemple de la notation de l'intensité.

La table 2 présente une typologie simple des éléments de notation musicale, principalement fondée sur leur étendue temporelle. Nous définissons également la notion *de tag courant* : pour un type de tag d'étendue temporelle implicite donné (clef, meter...), il s'agit de la dernière valeur rencontrée.

Basées sur cette typologie, des précautions sont à prendre lors des opérations suivantes :

- calcul du début d'une partition :

  1) tous les tags d'étendue temporelle explicite

étendue temporelle	description	exemple
explicite	la durée est explicite dans la notation	slur, cresc
implicite	la durée s'étend jusqu'au prochain signe similaire ou jusqu'à la fin	meter, dynamics, key
autres	contrôle de la structure	coda, da capo, repeats
-	instructions de formatage	new line, new page
-	notations diverses	breath mark, bar

Table 2. Typologie des éléments de notation.

doivent être ouverts (i.e. les tags à *début-ouvert*, voir section 3.2)

2) les tags courants d'étendue temporelle implicite doivent être rappelés,

– calcul de la fin d'une partition :

3) les tags d'étendue temporelle explicite doivent être fermés (i.e. les tags à *fin-ouverte*),

– mise en séquence de partitions :

4) les tags d'étendue temporelle implicite de la deuxième partition doivent être supprimés quand ils correspondent à un tag courant.

3.4. Cohérence de la structure

Les éléments qui relèvent de la catégorie *autres / contrôle de la structure* peuvent également donner lieu à des incohérences de notation : une barre de début de répétition sans barre de fin de répétition, un *dal segno* sans *segno*, un *da capo al fine* sans *fine*, etc. Nous introduisons de nouvelles règles pour la consistance des barres de répétitions. Nous définissons tout d'abord une barre de répétition *en suspens* comme le cas d'une barre de début de répétition sans barre de fin correspondante.

5) lors du calcul de la fin d'une partition, les barres de répétition *en suspens* doivent être fermées avec une barre de fin de répétition,

6) dans le cas de barres de début de répétition successives, seule la première sera retenue,

7) dans le cas de barres de fin de répétition successives, seule la dernière sera retenue.

Il n'y a pas de préconisation particulière pour les autres éléments de contrôle de la structure : les incohérences qui peuvent se produire sont ignorées mais ce choix préserve la réversibilité des opérations.

3.5. Réversibilité des opérations

Les règles définies ci-dessus résolvent la majorité des problèmes de notation mais ne permettent pas d'inverser les opérations : considérons une partition comportant une liaison qui est coupée en son milieu et rassemblée par une mise en séquences. Le résultat comportera deux liaisons (figure 6) en raison des règles **1)** et **3)** qui forcent l'ouverture des tags à *début-ouvert* et la fermeture des tags à *fin-ouverte*.

La résolution du problème requiert le support du langage GMN : nous introduisons un nouveau paramètre

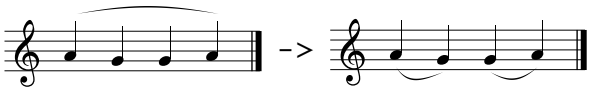


Figure 6. Une partition découpée et remise en séquence

pour les tags d'étendue temporelle explicite, conservant l'histoire du tag et indiquant un antécédent de type *fin-ouverte* et/ou *début-ouvert* . Ce paramètre est de la forme :

```
\tag<open="type">
```

où `type` est parmi [`begin`, `end`], correspondant respectivement à des antécédents *début-ouvert* et *fin-ouverte*.

Nous introduisons ensuite une nouvelle règle pour la composition de partitions, qui nécessite de définir des *tags adjacents* comme des tags placés sur la même voix et qui ne sont séparés par aucune note ou accord.

8) les tags *adjacents* similaires qui portent un paramètre `open` s'annulent mutuellement quand le premier est de type *fin-ouverte* et le second *début-ouvert*.

Ainsi, une opération rencontrant une forme de type :

```
\anytag<open="end">(f g)
\anytag<open="begin">(f e)
```

la transformera en :

```
\anytag(f g f e)
```

4. CONCLUSION

La complexité de la notation musicale repose en grande partie sur le grand nombre d'éléments de notation et sur leur statut hétérogène. La typologie que nous avons proposé (table 2) en est une simplification arbitraire, destinée à couvrir les besoins d'opérations de composition de partitions. Elle n'est pas représentative de cette complexité mais comme elle se base sur la sémantique de la notation, elle peut être appliquée à tout format de représentation de la notation musicale. Hormis les règles de réversibilité définies en 3.5 qui nécessitent le support du langage de représentation de la musique pour opérer, toutes les autres règles sont indépendantes du format GMN.

Les opérations sur les partitions peuvent se révéler très utiles dans le cas de traitements par lot (par exemple : séparation des voix d'un conducteur, extraction d'extraits, etc.). Les opérateurs présentés en table 1 permettent ce type de traitement, mais ouvrent également la voie à de nouvelles approches créatives de la musique.

5. REFERENCES

[1] H. Hoos, K. Hamel, K. Renz, and J. Kilian. The GUIDO Music Notation Format - a Novel Approach for Adequately Representing Score-level Music. In *Proceedings of the International Computer Music Conference*, pages 451–454. ICMA, 1998.

[2] H. H. Hoos and K. A. Hamel. The GUIDO Music Notation Format Specification - version 1.0, part 1 : Basic GUIDO. Technical report TI 20/97, Technische Universitat Darmstadt, 1997.

[3] Han-Wen Nienhuys and Jan Nieuwenhuizen. LilyPond, a system for automated music engraving. In *Proceedings of the XIV Colloquium on Musical Informatics (XIV CIM 2003)*, May 2003.

[4] Han-Wen Nienhuys. Lilypond, automated music formatting and the art of shipping. In *Forum Internacional Software Livre 2006 (FISL7.0)*, 2006.

[5] C. Daudin, D. Fober, S. Letz, and Y. Orlarey. The Guido Engine - a toolbox for music scores rendering. In *Proceedings of the Linux Audio Conference 2009*, pages 105–111, 2009.

[6] D. Fober, S. Letz, and Y. Orlarey. Open source tools for music representation and notation. In *Proceedings of the first Sound and Music Computing conference - SMC'04*, pages 91–95. IRCAM, 2004.

[7] Andreas Egler Daniel Taupin, Ross Mitchell. Musixtex using tex to write polyphonic or instrumental music.

Index des auteurs

WWW.JIM2012.BE

UMONS
Université de Mons

Lomus 2012