Étude et implémentation de procédures de contrôle « continu » pour la synthèse par fonctions d'ondes formantiques

Raphaël Foulon

Mars-Juillet 2012

IRCAM, équipe Représentations Musicales,
sous la direction de Jean Bresson

Remerciements

En préambule à ce mémoire, je tiens à remercier, pour sa confiance, sa patience, sa disponibilité et sa sympathie, Jean Bresson, grâce à qui j'ai pu effectuer un stage riche en enseignements, dans une ambiance agréable.
J'adresse un grand merci à Marco Stroppa et à Carlo Laurenzi, pour leur écoute et leur bonne humeur.
Merci à Carlos Agon pour l'aide qu'il a pu apporter durant toute cette année.
Merci à Xavier Rodet, dont l'expérience et les conseils ont été très précieux.
Merci à Philippe Cuvillier et à Léa Franchon, avec qui j'ai partagé d'agréables moments, du haut du quatrième étage de l'IRCAM.
J'aimerais aussi remercier les étudiants ATIAM, avec qui j'ai pu passer une année riche et épanouissante.

Enfin, j'aimerais remercier les personnes qui ont été cette année pour moi une inspiration, tant du point de vue scientifique qu'artistique, en particulier Evelyne Gayou et Michèle Castellengo.

Résumé

Ce travail concerne le contrôle de la synthèse par « fonctions d'ondes formantiques », et plus particulièrement du synthétiseur CHANT, dans le cadre de processus compositionnels. Lors de ce stage, nous avons travaillé dans l'environnement de composition assistée par ordinateur OpenMusic, et mis en place des procédures permettant de contrôler les transitions et superpositions entre « événements » sonores produits par le synthétiseur.

Ces processus nous ont permis de modéliser des phénomènes liés au modèle de la voix, notamment par l'étude des consonnes et autres « articulations » vocales, mais aussi de dépasser ce modèle pour produire des sons d'une grande diversité et en proposer des modes de contrôles expressifs.

Les problématiques traitées dans ce travail concernent l'analyse et la modélisation des phénomènes sonores (et vocaux en particulier), la conception de méthodes et d'interfaces adressées aux compositeurs, et s'étendent d'un point de vue théorique aux questions plus générales liées à la dualité entre objets symboliques (musicaux) et signaux, et plus particulièrement entre évènements discrets et phénomènes continus.

Mots-clés : CHANT, synthèse par règles, contrôle de la voix chantée, composition assistée par ordinateur

Abstract

This work is about the control of FOF synthesis (FOF stands for "fonctions d'ondes formantiques"), and especially the CHANT synthesizer, for compositional purposes. During this internship ,we have worked with OpenMusic, a computer assisted composition environment, and designed procedures which may control the transitions and the overlaps between sound "events" produced by the synthesizer.

These processes allowed us to imitate phenomenons linked to the vocal model, especially with the study of consonants and other vocal "articulations", but also to go beyond this model to produce a great variety of sounds and to offer a means of expressive control.

The problematics dealt with this work are centered on the analysis and the modelling of sound phenomenons (especially vocal ones), the creation of methods and interfaces specifically for composers, and from a theorical point of view to more general questions linked to the duality between symbolic (musical) objects and signals, and especially between discrete events and continuous phenomenons.

Keywords : CHANT, rule synthesis, singing voice control, computer-aided composition

Table des matières

1 Présentation du sujet

La synthèse par fonctions d'ondes formantiques (FOF), et en particulier le synthétiseur CHANT, ont connu un essor dans les années 80, pour ensuite être marginalisés, au fur et à mesure que la longueur des tables d'ondes et la cadence des processeurs augmentaient. Basée sur le modèle de production vocale, cette technique permet la production de timbres voisés de très haute qualité, et peut être aisément étendue à de tous autres types de sons. Un système de contrôle de CHANT a été ré-implémenté récemment dans l'environnement de CAO OpenMusic, permettant de structurer les différents paramètres de la synthèse (fréquence fondamentale, fréquence, amplitude, largeur de bande des formants...) qui seront transmis au synthétiseur.

Le système CHANT a pour spécificité de présenter une logique de contrôle continue : contrairement à des protocoles plus courants tels que Csound ou MIDI, celui-ci s'effectue via un ensemble de paramètres spécifiés à des instants donnés, qui seront interpolés temporellement avant d'être intégrés dans un processus de synthèse. Les protocoles de contrôle définis dans OpenMusic, basés sur la notion d'évènements (éléments atomiques de contrôle, localisés et étendus dans le temps), rendent cependant au contrôle un caractère discret, plus naturel dans les environnements symboliques et les processus compositionnels.

L'objectif principal de ce stage consiste à définir et implémenter des procédures de contrôle « continu » pour le synthétiseur CHANT, afin d'agir avec précision sur la production sonore, dans ce contexte compositionnel et au sein de cette logique évènementielle. Nous cherchons à définir avec précision et expressivité les comportements du système lors de la transition entre le domaine discret (OpenMusic) et continu (CHANT). Il s'agit d'élaborer un ensemble de « règles », qui permettront de générer une suite de paramètres correspondant à un effet désiré (production d'une consonne, d'un vibrato...) ; chaque règle ayant pour but de formaliser un aspect spécifique du processus de synthèse, sur une certaine durée.

Ce stage a été réalisé à l'IRCAM, au sein de l'équipe « Représentations Musicales ». Cette équipe se spécialise dans la représentation symbolique des structures musicales de haut niveau. Ses principaux thèmes de recherche comprennent la musicologie computationnelle et la composition assistée par ordinateur (CAO). Elle participe en particulier au développement et à la diffusion du logiciel OpenMusic, utilisé par de nombreux compositeurs à travers le monde, qui est la référence en matière de système de CAO.

2 État de l'art

2.1. Faire chanter l'ordinateur

2.1.1 CHANT et les fonctions d'ondes formantiques

La synthèse par FOF s'inspire du modèle de production de la voix, et permet de reproduire – par exemple – le comportement des résonateurs de l'appareil vocal. Cette méthode de synthèse consiste à générer une ou plusieurs sinusoïdes amorties, à chaque « tic » donné par un train d'impulsions (dont la fréquence fondamentale, ou *f0*, peut être paramétrée). Ces sinusoïdes respectent l'équation (1) :

$$s(n) = \begin{cases} 0 & n \leq 0 \\ \frac{1}{2}(1 - cos(\beta n))e^{-\alpha n} sin(\omega_c + \phi) & 0 \leq n \leq \frac{\pi}{\beta} \\ e^{-\alpha n} sin(\omega_c n + \phi) & n > \frac{\pi}{\beta} \end{cases}$$

eq. (1) [1]

Chaque train de sinusoïdes contribue au spectre sonore par un *formant*. Les *formants* sont les modulations spectrales caractéristiques du signal vocal. En d'autres termes, ils correspondent à des zones du spectre riches en intensité, identifiables par exemple sur la figure 1.

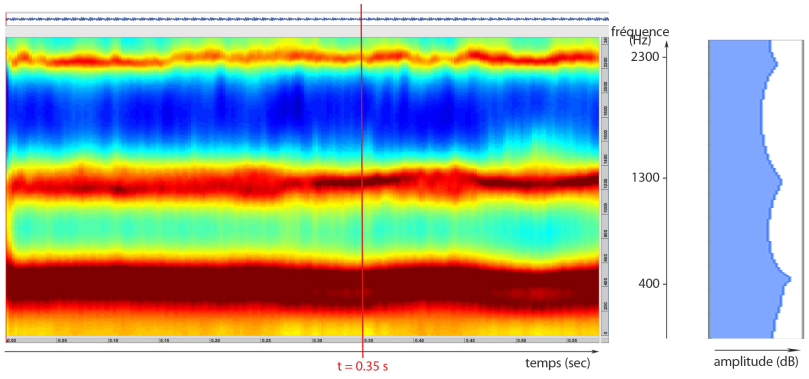

Figure 1 – A gauche, analyse d'un extrait de voix, sur lequel on peut identifier trois formants, dont les fréquences centrales sont respectivement 400 Hz, 1300 Hz et 2300 Hz. A droite, spectre instantané obtenu à t=0.35s.

Le nombre de formants présents dans le spectre de sortie sera égal au nombre de sinusoïdes générées à chaque impulsion, soit au nombre de générateurs de FOF mis en parallèle dans la

1. Les paramètres de cette équation seront expliqués plus loin.

synthèse. Leurs caractéristiques spectrales seront directement liées aux paramètres de ces sinusoïdes [Rodet, 1984] (cf. figure 3). Ainsi, il est possible de modeler le spectre des formants en modifiant les paramètres des FOF : dans l'équation 1, la fréquence de résonance ω_c détermine la fréquence centrale du formant, α et β déterminent la largeur de bande à -40dB [2] (aussi appelée largeur de la jupe) et la largeur de bande à -3dB (cf. figure 2).

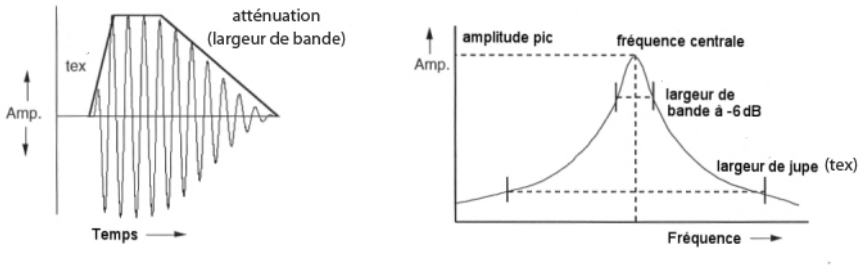

Figure 2 – Vue parallèle de la forme d'onde d'une FOF et des caractéristiques spectrales du formant associé. Illustration extraite du cours de synthèse vocale par Caroline A. Traube et Olivier Belanger, source de l'image disponible sur http ://cours.musique.umontreal.ca/mus1321/Notes_de_cours/MUS1321_synthese_vocale.pdf

Le synthétiseur CHANT a été implémenté par Xavier Rodet et son équipe à l'IRCAM dans les années 1980 [Rodet et al., 1984]. Étant donné le rapport direct des équations de FOF aux paramètres spectraux des formants, le synthétiseur donne la possibilité de contrôler la synthèse directement à partir de ces paramètres, soit la fréquence, l'amplitude du pic (amplitudes relatives entre plusieurs sinusoïdes associées à une même impulsion), la largeur de bande à -6dB, le temps d'excitation de la sinusoïde, qui détermine la largeur des jupes, et la phase.

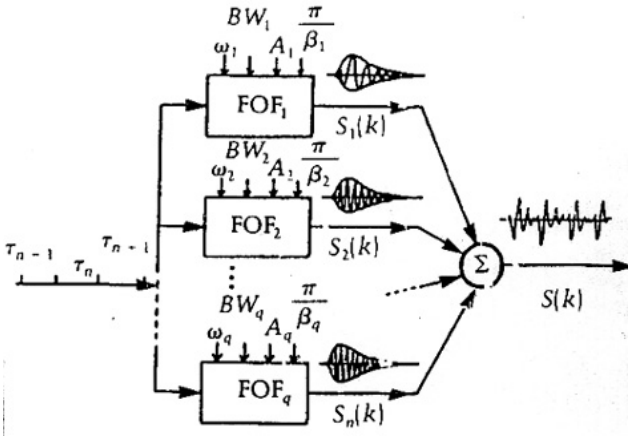

Figure 3 – Schéma du synthétiseur CHANT [Rodet et al., 1984]

2. La référence étant le maximum de la FOF.

Comme nous l'avons mentionné en introduction, CHANT propose un modèle de contrôle « continu » axé sur la notion de « phrases », dans lequel le synthétiseur réalise par interpolation la transition entre les états et valeurs des paramètres du synthétiseur spécifiés à intervalles plus ou moins réguliers par le système ou processus de contrôle.

Implémentation de référence de la synthèse par FOF, le synthétiseur était associé à l'environnement Formes, un logiciel de composition assistée par ordinateur, basé sur le langage Lisp, et qui permettait de générer ces « phrases » via la programmation et l'ordonnancement de processus temporels [Rodet et Cointe, 1984]. Les processus créés dans Formes avaient pour vocation de contrôler l'évolution des paramètres de synthèse, et notamment d'activer et paramétrer un ensemble de règles codées dans le synthétiseur, permettant de modifier ces paramètres, par exemple afin d'obtenir un vibrato, de corriger le spectre du signal vocal (en fonction de l'effort vocal souhaité, du type de voix, etc...) [Baisnee, 1985], ou encore de créer des « consonnes » entre les voyelles synthétisées [Rodet et Depalle, 1985].

2.1.2 La production de phonèmes, approche par concaténation et par règles

Différentes techniques permettent de simuler la voix, et plus particulièrement la voix chantée. Par exemple, la synthèse par concaténation de diphones synthétisés ou échantillonnés, permet de produire des phrases vocales d'un grand réalisme. Nous pouvons citer en particulier, le programme *Diphone Studio* [Rodet et Lefevre, 1997], qui propose de construire des phrases vocales en effectuant des morphings entre plusieurs phonèmes, obtenus par échantillonnage ou produits par différents systèmes de synthèse, dont CHANT.

Une autre approche est l'utilisation de règles, qui, élaborées suite à l'étude du fonctionnement et des comportements des signaux vocaux et de leurs modes de production, permettent de les reproduire au sein de processus de synthèse. On peut en particulier citer les travaux de Johan Sundberg [Sundberg, 1987], qui représentent des sources importantes pour l'implémentation de procédures de contrôles fines et expressives pour un synthétiseur comme CHANT.

De précédents travaux ont cherché à exploiter les résultats de telles analyses en vue de la reproduction de phénomènes vocaux à l'aide de règles. Le synthétiseur MUSSE en est un exemple : initialement sous forme analogique puis logicielle, il propose d'effectuer une conversion texte vers chant, tout en introduisant de nombreuses règles (vibrato, consonnes, intonation...), la mélodie étant contrôlable en temps réel (grâce à un clavier) [Sundberg, 2006].

Néanmoins, comme nous le verrons plus loin, notre but ici ne se réduit pas à la reproduction fidèle du jeu d'un chanteur : nous chercherons à nous inspirer des modes de production vocale dans le but de « dépasser » les propriétés traditionnelles de la synthèse vocale.

2.2. Composition assistée par ordinateur

2.2.1 OpenMusic, OM-Chant

En opposition au workflow habituel des environnements musicaux orientés vers le traitement du signal, les systèmes de composition assistée par ordinateur (ou CAO) [Assayag 1998] proposent une

approche originale axée sur la manipulation des données compositionnelles symboliques (représentations de partitions, suites de notes, structures rythmiques) au sein de langages de programmation dediés. Cette spécificité leur permet d'offrir aux compositeurs une grande puissance au niveau de l'expressivité, ainsi que la possibilité de conserver la trace d'une pensée musicale structurée.

Parmi ces environnements, OpenMusic est un langage de programmation visuelle basé sur Common Lisp, développé à l'IRCAM, et utilisé par de nombreux compositeurs, conservatoires et universités à travers le monde [Assayag et al. 1999, Agon 1998, Bresson et al. 2009].

Parmi de précédents travaux portés sur l'intégration des processus de synthèse sonore dans le cadre d'environnements de composition assistée par ordinateur, plusieurs projets ont par le passé visé à contrôler le synthétiseur CHANT, développés dans un premier temps dans le langage Patchwork (avec notamment la bibliothèque PW-Chant de L. Pottier), puis dans OpenMusic, son successeur, avec les bibliothèques Chant-lib ou OM-Chant [Bresson et Michon, 2011]. Dans OM-Chant notamment, quelques-unes des règles de contrôles du système CHANT/Formes ont été portées dans OpenMusic.

2.2.2 Contrôle discret et continu

Des objets sonores de haut niveau définis dans OpenMusic sous forme de « matrices » de paramètres pour les différents formants (ou autres composants de contrôle) structurent les paramètres en tant qu'« évènements » de synthèse [Bresson et Stroppa, 2011]. Ces événements de synthèse peuvent être des valeurs de fréquence fondamentale du train d'impulsions, ou f0, des matrices de paramètres de FOF, des générateurs de bruit ou des filtres formantiques.

Dans l'implémentation actuelle de CHANT, le synthétiseur est contrôlé grâce à une suite de *frames* SDIF [3], éléments atomiques du contrôle, déterminant l'état du synthétiseur à un moment donné. Chaque événement produit un certain nombre de *frames*, qui sont ensuite ordonnées et interpolées lors du processus de synthèse, pour obtenir un flux de contrôle continu. Si les paramètres sont stables (constants sur la durée de l'événement) il y a une frames au début et une à la fin, sinon, l'événement est décrit par une séquence de frames décrivant l'évolution des paramètres.

Dans ce contexte, qui est le point de départ de notre travail, les comportements pouvant apparaître en cas d'événements de contrôle distants, contigus ou superposés restent à concevoir.

En effet, des « zones d'ombre » apparaissent lorsque vient le moment de créer un flux sonore continu à partir des événements discrets. La figure 4 illustre l'un des problèmes obtenus lors de la superposition de deux de ces événements, qui résulte en une aberration au niveau de la continuité de la synthèse.

3. Le format SDIF permet d'encapsuler des descripteurs sonores dans un format standardisé (Sound Description Interchange Format) [Wright et al., 1999].

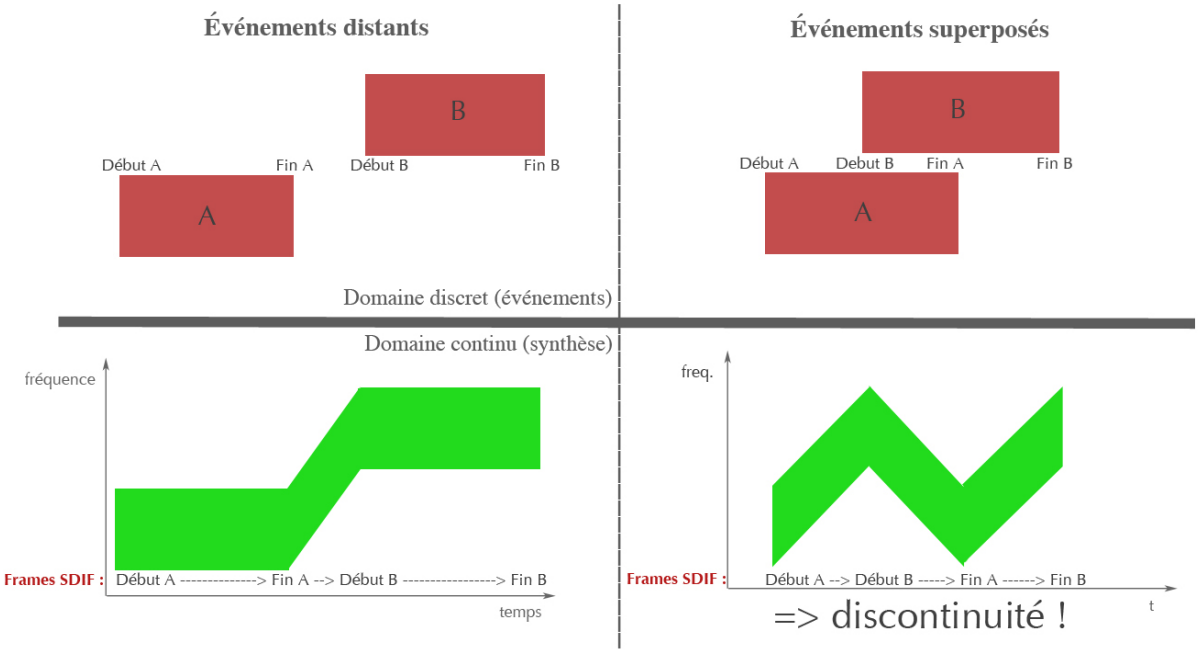

Figure 4 – Illustration schématique du problème de la traduction d'événements superposés : chaque événement envoie au synthétiseur deux frames SDIF, qui, après qu'elles soient ordonnées, créent une « aberration » dans le processus de synthèse.

Des solutions au problème du passage de l'écriture événementielle au contrôle continu ont été proposées dans la littérature : nous pouvons citer par exemple la technique de « *procedural concatenation* » [Anderson et Kuivila, 1989], qui consiste à modéliser des phénomènes continus par concaténation de fonctions élémentaires. Bien qu'appliquée à la synthèse temps réel, cette méthode propose des pistes intéressantes quant à la gestion du temps et des cadences d'appel des procédures de contrôle.

3 Stratégies de contrôle, analyse et extraction des phénomènes vocaux

Avant d'implémenter de nouvelles procédures de contrôle de CHANT, nous avons effectué une étude des phénomènes de production vocale, à partir d'analyses temps-fréquence d'extraits audio. Ce travail préliminaire nous permettra de reproduire certains de ces phénomènes et de les modifier, en effectuant une modélisation spécifique à la synthèse par FOF.

3.1. Méthode d'analyse des signaux de voix chantée

Les analyses que nous avons effectué ont été obtenues grâce au logiciel Audiosculpt [Bogaards et Röbel, 2004]. Les analyses temps-fréquence ont été produites en utilisant la méthode LPC (Linear Predictive Coding), qui est particulièrement efficace dans le cadre des signaux de voix. En effet, cette technique se base sur un modèle de prédiction linéaire, qui approxime le comportement de l'appareil vocal grâce à un modèle source-filtre [Li et al, 2003].

Le support et la collaboration de Xavier Rodet, de l'équipe *analyse-synthèse* de l'IRCAM, nous ont permis de déterminer des paramètres efficaces pour l'analyse. Nous avons utilisé une prédiction linéaire d'ordre important (autour de 100) : en effet, on néglige la contribution de la composante bruitée dans le cadre des sons voisés [4]. Les meilleurs résultats ont été obtenus avec de grandes valeurs de suréchantillonage : 8x pour une fenêtre d'analyse de 1024 échantillons, et un suréchantillonage de FFT de 4x. Le tout, en faisant varier la fréquence fondamentale en fonction des données à analyser. Enfin, la visualisation des sonogrammes de type « hot-cold » [5] (voir figure 5), nous a permis d'observer nettement mieux les évolutions formantiques de faible durée (de l'ordre de 50ms).

4. Il faudrait réduire l'ordre de la LPC pour l'étude de sons non-voisés avec composante bruitée.
5. Les couleurs variant en fonction de l'intensité, du bleu au rouge.

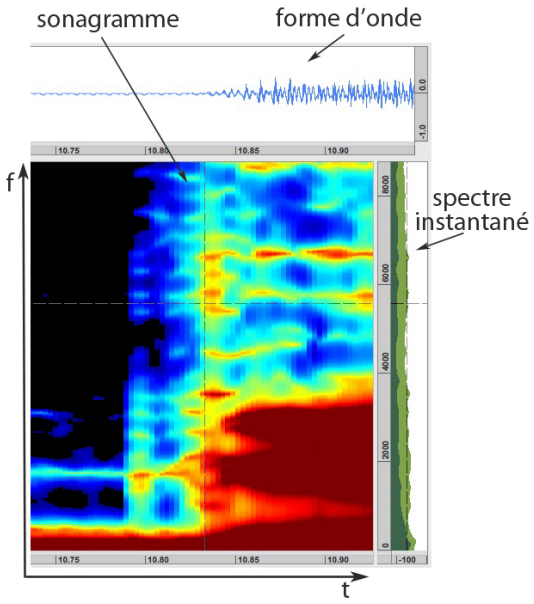

Figure 5 – Analyse du diphone 'gɜ' effectuée dans Audiosculpt, avec affichage *hot-cold* pour les amplitudes.

De multiples sources ont été mises à profit pour trouver des échantillons sonores pertinents. Dans un premier temps, nous avons récupéré une ancienne base de données audio créée en 1983 par Xavier Rodet. D'une durée totale de deux heures, elle présente l'enregistrement d'un ténor d'opéra, chantant une large combinaison de phonèmes (triplets voyelle-consonne-voyelle, et quadruplets voyelle-consonne-consonne-voyelle). Elle fut très utile pour observer les phénomènes liés au langage, et pour modéliser un premier jeu de consonnes au sein de notre système.

Des enregistrements d'opéras célèbres (en particulier, des passages de chant staccato et legato de « La Reine de la Nuit » de Mozart) ont aussi été analysés [6].

Des extraits sonores de la pièce « Chréode » de Jean-Baptiste Barrière, réalisée à l'IRCAM en 1983 [7], produits grâce à une précédente implémentation du synthétiseur CHANT au sein de l'environnement Formes, ont aussi été étudiés. En effet, bien que les paramètres de contrôle de la synthèse aient été perdus depuis sa création, ces archives témoignent d'une utilisation originale de la synthèse par FOF.

Enfin, nous avons procédé à plusieurs enregistrements de voix, avec du matériel semi-professionnel (microphone à condensateur large membrane Sontronics STC-2 et carte son M-Audio Fast Track Ultra) afin d'analyser certains phénomènes particuliers, comme la production de consonnes à des « vitesses d'élocution » différentes.

6. Bien que la piste sonore comprenne un accompagnement instrumental, le fait que l'orchestre ne joue pas au dessus de la nuance piano a permis de réaliser des analyses pertinentes.

7. Cette pièce a reçu le Prix de la Musique Numérique du concours international de musique électro-acoustique de Bourges en 1983.

3.2. Modèle utilisé

L'étude d'extraits de voix (phrases de chant lyrique, de parole, de production de phonèmes) a permis d'aboutir à une modélisation élémentaire, ne considérant que sa composante voisée. Dans ce modèle, nous dégageons deux phénomènes : les états pseudo-stables et les *transitions*[8]. Cette modélisation s'inspire des travaux effectués lors de l'implémentation de règles pour le contrôle de CHANT dans l'environnement *Formes*. Le critère de discrimination utilisé pour définir les deux phénomènes est basé sur le comportement des formants (on ne prend pas en compte la f0, qui dans CHANT est gérée indépendamment des paramètres de FOF). Le sonagramme visible sur la figure 6 présente un exemple sur lequel ces deux états sont identifiés :

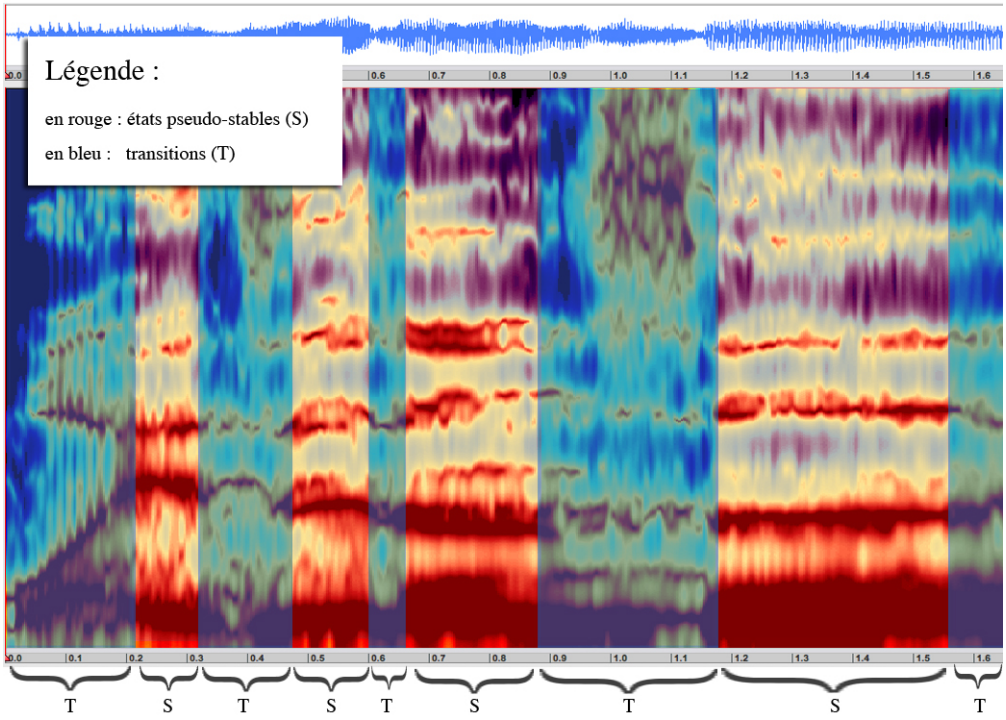

Figure 6 – Analyse d'un extrait de voix chantée, identification des états pseudo-stables (S) et des transitions (T).

3.2.1 Etats (pseudo) stables

Nous définissons un état stable comme un état continu dans le temps, au sein duquel les paramètres formantiques n'évoluent pas ou peu : pour donner un ordre de grandeur, les variations ne doivent pas excéder 10 % pour les fréquences centrales, de 3dB pour les amplitudes et de 20% pour les largeurs de bande.

Dans le cadre de la voix parlée, les états stables peuvent être assimilés aux voyelles, ou dans le cas du chant lyrique, aux notes tenues. Cependant, les parties centrales de certaines consonnes

8. Dans la suite de ce rapport, nous simplifierons la dénomination « état pseudo-stable » à « état stable ».

voisées, comme le 'm' ou le 'l' pourront également être modélisées par un état stable, du fait de la faible variation des formants en leur centre. Nous reviendrons sur ce point dans la partie 3.2.3, qui présente quelques cas particuliers de modélisation.

En synthèse CHANT, ces états pourront être produits par des FOF aux paramètres formantiques constants. Cependant, cette approximation ne prend pas en compte les petites variations temps-fréquence qui donnent son « caractère » à la voix : si l'on ne fait que générer une suite de FOF avec un train d'impulsions de f0 constante, le résultat sonore est perçu comme très artificiel. C'est pourquoi nous parlons ici d'une *pseudo*-stabilité. Pour obtenir une voix plus naturelle, il faut reproduire une certaine quantité de variations dans les paramètres formantiques. Deux solutions ont été abordées et testées avec succès (nous détaillerons leur implémentation plus loin) :

- variations de la f0 : on conserve des paramètres de FOF constants, mais on module la f0 avec, par exemple, un *jitter*[9], ou un vibrato, ce qui est une solution particulièrement adaptée au chant d'opéra.
- variations aléatoires des FOF : introduction d'aléatoire dans la fréquence, l'amplitude et la largeur de bande, ce qui transforme une FOF constante en une FOF paramétrée par des fonctions temporelles.

3.2.2 Transitions et trajets formantiques

Comme visible sur la figure 6, et par opposition aux états stables, les *transitions* sont les parties du spectre dans lesquelles les formants présentent une évolution significative.

Pour effectuer une analogie avec la partie précédente, les *transitions* peuvent être assimilées aux consonnes (mais cela est aussi réducteur lorsqu'on prend en compte les consonnes voisées), ou aux articulations en chant lyrique (staccato, légato, etc...).

Alors que les états pseudo-stables peuvent être représentés par des valeurs de paramètres de FOF (constants ou évoluant en fonction du temps), les *transitions* seront définies à l'aide de ce que nous appellerons des *trajets formantiques*. Ces trajets permettent de définir le comportement temporel des fréquences, amplitudes, largeurs de bande (voire d'autres paramètres, tels que le temps d'excitation [10]) entre deux états stables.

Nous avons élaboré un certain nombre de trajets formantiques à partir d'observations de signaux, qui seront ensuite généralisés à l'interpolation (des paramètres formantiques) entre les deux états stables qu'ils lient.

Afin d'illustrer ce concept de trajet formantique, nous allons décrire la procédure de contrôle mise en place lors de la reproduction de la consonne 'b', dans la syllabe 'ɔbɔ' (seul le paramètre de la fréquence centrale du premier formant est ici pris en compte, pour simplification), sur la figure 7.

9. Un jitter est une variation périodique aléatoire autour d'une valeur.
10. Rappelons que la forme de l'enveloppe d'amortissement des sinusoïdes détermine la forme spectrale des formants.

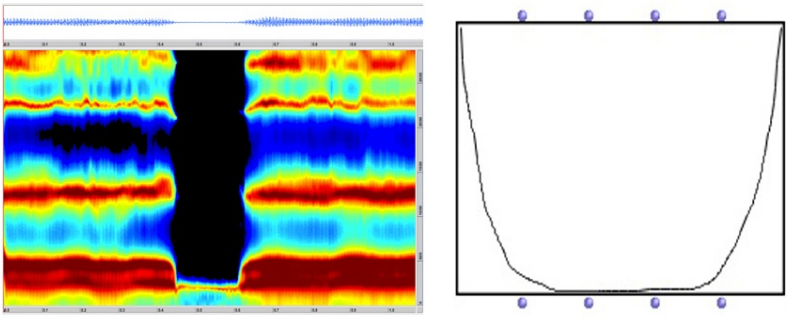

Figure 7 – En haut, de gauche à droite : analyse de la syllabe ‘əbə’, et trajet formantique s’appliquant à la fréquence centrale du premier formant dans la *transition* ‘b’.

Le processus illustré sur les figure 7 et 8 peut être décrit ainsi : dans un premier temps, un *morphing* est effectué entre les deux états stables jouxtant la *transition*. En d’autres termes, le paramètre étudié ici est interpolé linéairement. Puis, on applique à ce morphing le trajet formantique subi par l’amplitude du premier formant, par multiplication. Ce trajet formantique a été élaboré à partir d’analyses de consonnes ‘b’, et en particulier l’observation du comportement de l’amplitude du premier formant (qui effectue donc une courbe convexe). On obtient alors le trajet effectué par l’amplitude du premier formant au cours de la génération de la consonne ‘b’, dans la syllabe ‘iba’. Pour modéliser correctement une transition, il faut répéter ce processus pour chaque formant, et pour chacun de ses paramètres. Le sonagramme du résultat est visible sur la figure 8.

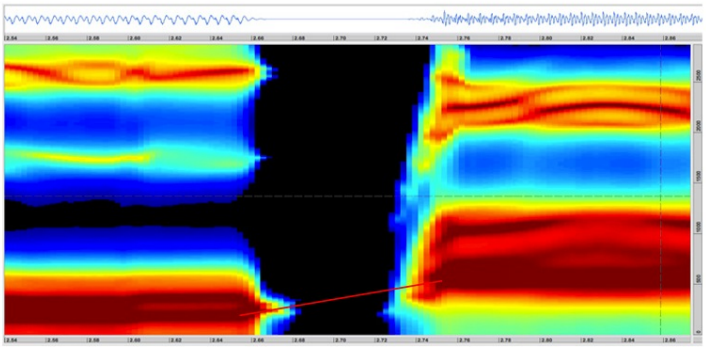

Figure 8 – En bas : sonagramme de la syllabe ‘iba’, obtenue par synthèse après application des différents trajets formantiques.

Pour obtenir un contrôle plus fin sur le trajet formantique, nous introduisons un paramètre supplémentaire : la courbe d’interpolation. Dans l’exemple précédent, le trajet formantique s’applique à un morphing entre le premier et le second état stable. Cependant, il peut s’avérer utile d’effectuer une interpolation non linéaire entre les deux états stables : dans l’exemple de la consonne ‘b’, un morphing hyperbolique donnera de meilleurs résultats :

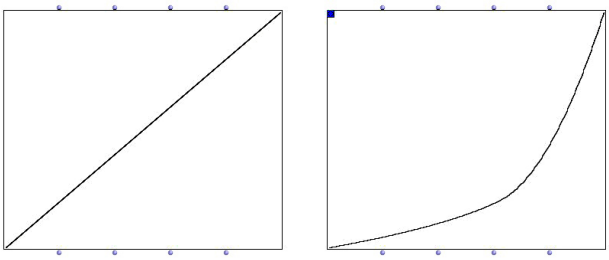

Figure 9 – A gauche : fonction de morphing linéaire, à droite, fonction de morphing hyperbolique, plus adaptée aux consonnes.

Nous appellerons dans la suite l'ensemble (trajets formantiques / courbe de morphing) le *profil formantique* d'une transition.

Définir les différents phénomènes vocaux (syllabes, groupes phonème-articulation, etc...) par leur trajet permet d'aborder le problème du contrôle de la synthèse de manière modulaire, contrairement à d'autres techniques comme la concaténation de diphones, dans laquelle une consonne est définie de manière statique, en fonction du triphone [11] dans laquelle elle est présente. C'est ce qui fait ici la force de cette méthode, à mi-chemin entre synthèse concaténation (succession d'états stables et de transitions) et par règles (application de trajets formantiques).

Nous pouvons noter cependant, qu'une approximation importante a été faite, en ce qui concerne la voix : nous considérons ici que le comportement d'une transition ne dépend que de trois paramètres : le profil formantique, et les états pseudo-stables précédant et suivant la transition. Or, il a été démontré que par exemple, dans le cas de la voix parlée, la production d'une consonne peut dépendre de la voyelle précédant le triplet voyelle-consonne-voyelle concerné [Rodet et Depalle, 1985].

3.2.3 Cas particuliers liés à la synthèse vocale

Comme nous l'avons introduit plus tôt, on ne peut classer les phonèmes, au sens donné par la phonétique, strictement dans les deux catégories énoncées ci-dessus, et des cas particuliers sont à aborder.

Nous pouvons prendre l'exemple du 'm' [12], qui peut s'appliquer à d'autres phonèmes tels que le 'n', le 'l', ou même le 'ʒ' et le 'z', si l'on ne considère que leurs composantes voisées.

En observant le spectre de la syllabe 'ama', on peut clairement identifier cinq parties (cf. figure 10) :

- les deux voyelles 'a'
- ce qu'on pourra appeler le « fade in » et le « fade out » du 'm'
- la partie centrale du 'm', présentant peu de variations formantiques

11. Un triphone est un triplet voyelle-consonne-voyelle.
12. 'm' est une consonne dite occlusive nasale bilabiale voisée.

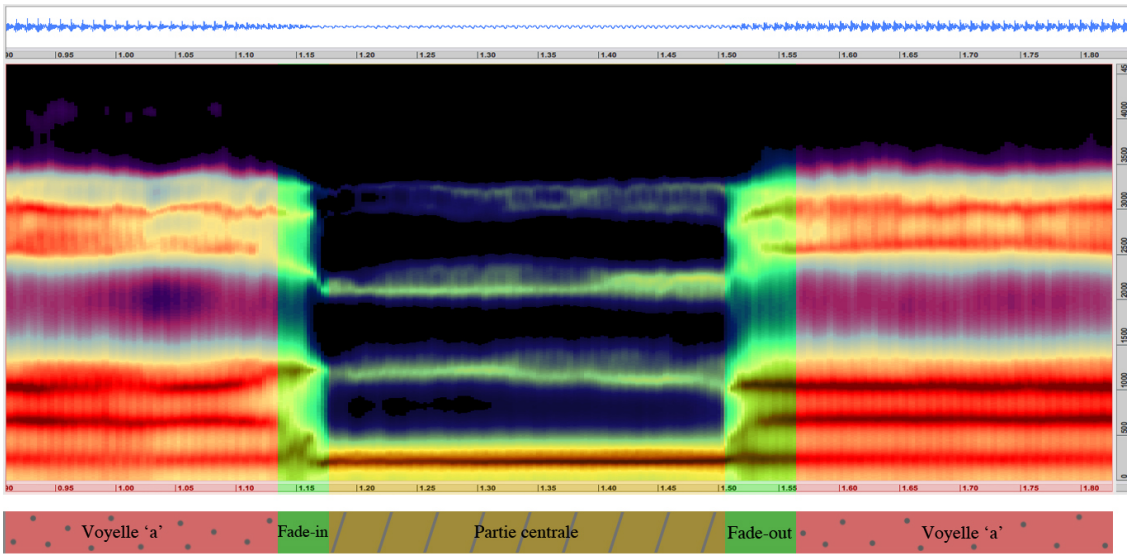

Figure 10 – Analyse du triphone ‘ama’, et identification des trois parties remarquables.

On peut, à partir de cette observation, procéder à deux modélisations différentes :

La première consiste à ne définir le ‘m’ que grâce à une *transition*. Les trajets formantiques la définissant suivront alors la courbe décrite par le *fade-in*, le *fade-out* et la partie centrale. En pratique, le modèle permet de produire un ‘m’ réaliste si l’on considère des consonnes dont la durée n’excède pas les 200ms. Mais au delà de cette durée, les parties du trajet correspondant aux fade-in et fade-out se retrouveront trop étirées dans le temps, ce qui produira un rendu sonore qui pourra être jugé artificiel. Néanmoins, cet effet peut être mis à profit dans un contexte artistique.

La deuxième méthode propose de ne pas étirer les fade-in et fade-out, en modélisant le triphone ‘ama’ à l’aide de trois états stables : deux pour les voyelles et un pour la partie centrale (qui sera étirable dans le temps), et deux *transitions* correspondant aux fade-in et fade-out, dont les durées pourront être fixées, indépendamment de la durée effective de la consonne.

Par souci de concision, nous n’allons pas détailler le traitement de chaque cas particulier, cependant, nous pouvons en évoquer quelques uns :

- débuts et fins de phrases : ils pourront être représentés par des états stables ayant des paramètres d’amplitude nuls (silencieux), suivis d’une *transition*.
- le ‘r’ « roulé » : il s’agit d’un son pseudo-périodique, avec un pattern ‘état stable - transition’ répété. La modélisation puis l’implémentation de la procédure de contrôle correspondante seront détaillées dans la partie 5.2.1.

4 Une première implémentation du contrôle des transitions avec OM-Chant

4.1. Rappels généraux sur l'environnement OpenMusic/CHANT

L'environnement de CAO OpenMusic permet de construire des programmes visuels, appelés « patches » (un exemple de patch est visible sur la figure 11). L'interface se présente sous la forme de boîtes dotées d'entrées et/ou de sorties, représentant des instances de classes ou des fonctions, que l'on peut relier entre elles par des connections, pour former un graphe dataflow.

Comme nous l'avons introduit dans l'état de l'art, la bibliothèque OM-Chant propose plusieurs objets, dotés d'attributs temporels tels qu'une date de début et une durée [13], représentant des événements de contrôle pour le synthétiseur CHANT. Ces objets permettent de formaliser les processus de synthèse, à partir d'évènements de synthèse dissociés et contrôlant différents paramètres, tels que :

- des valeurs de f0 (scalaires ou fonctions) [objets de type *ch-f0*] ;
- des paramètres formantiques (matrices de n formants) [objets de type *ch-FOF*] ;
- des générateurs de bruit [objets de type ch-noise] ;
- des filtres formantiques [14] [objets de type ch-FLT].

Ces différents objets permettent de régler les paramètres de synthèse et leur évolution dans le temps. Les événements sont représentés par des matrices, dont les colonnes contiennent les différents « composants » de synthèse (ici, les différents générateurs de FOFs produisant chaque formant). Dans chaque « case » de ces matrices se trouve une valeur constante sur la durée de l'événement, ou une courbe représentant une évolution du paramètre idoine.

Lorsque deux événements se suivent (qu'ils soient de type ch-FOF ou ch-f0), une interpolation linéaire est effectuée entre les valeurs des paramètres à la fin du premier et au début du second événement. Un exemple est donné à la figure 11.

Pour plus de précisions concernant le contrôle de CHANT dans l'environnement OpenMusic, on pourra se référer à [Bresson et Stroppa, 2011].

13. Ainsi que des durées de fade-in et de fade-out, que nous n'utiliserons pas ici.

14. Ces deux derniers éléments permettent d'ajouter des composantes bruitées, ou d'effectuer de la synthèse soustractive à base de filtres résonnants, au sein des procéssus de synthèse. Cependant, nous ne les utiliserons pas ici, puisque nous nous restreignons à l'étude des sons voisés.

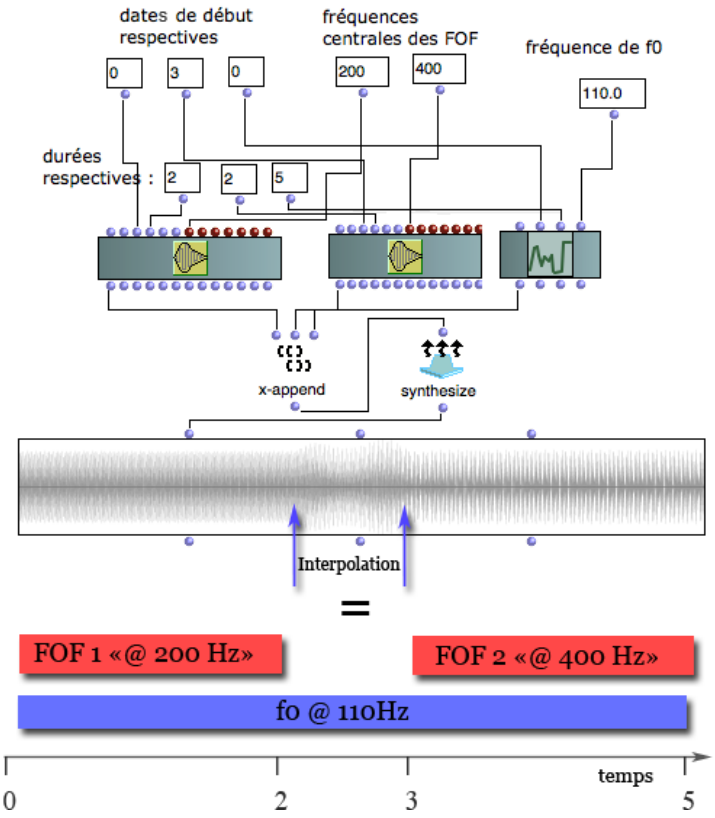

Figure 11 – Patch et illustration symbolique d'un processus de synthèse, faisant intervenir deux matrices de FOF (contenant chacune un formant) et un objet f0.

4.2. Gestion séquentielle des événements

Une fois la phase de modélisation effectuée, nous avons implémenté une première version d'outils permettant de mettre en place des procédures de contrôle d'une phrase de voix chantée avec OM-Chant. En particulier, nous souhaitons traiter les cas d'événements distants, en les considérant comme des états stables, et générant les *transitions* qui les séparent.

Dans la bibliothèque OM-Chant, un outil permet de traiter les transitions : il s'agit de la fonction *ch-transitions*. Cette fonction permet de gérer et de transformer une séquence d'objets *ch-FOF*, traitant chaque couple d'événements successifs de manière itérative, en leur appliquant une une fonction [15], elle aussi donnée en entrée [16]. La figure 12 propose un exemple d'une telle fonction, sous forme de patch, permettant d'implémenter des solutions au problème des superpositions.

15. Une *lambda-fonction*, propre au paradigme fonctionnel.
16. *ch-transitions*, qui reçoit une fonction en tant que paramètre, est aussi appelée « fonction d'ordre supérieur ».

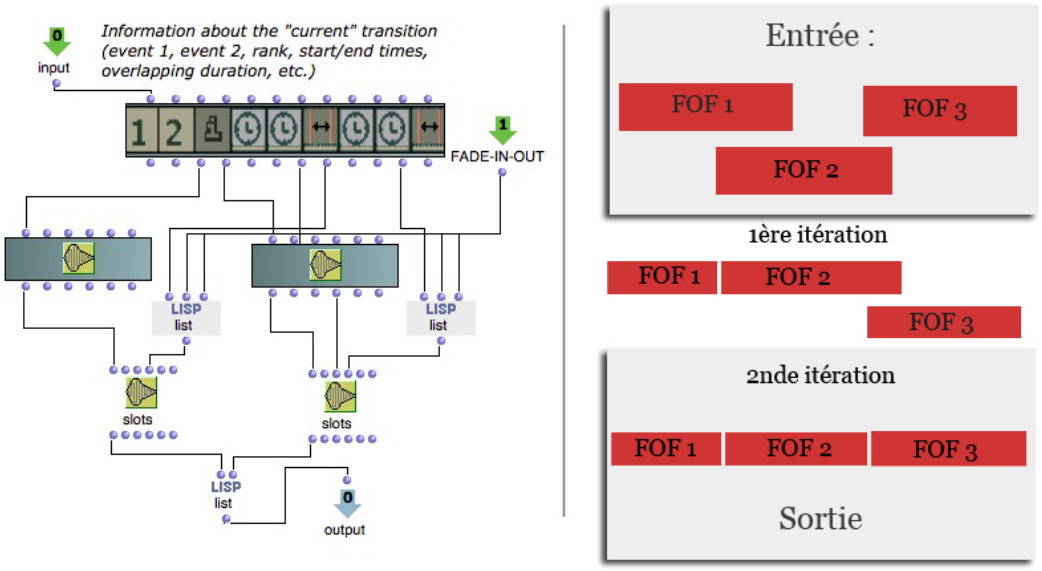

Figure 12 – A gauche : patch de la fonction permettant de gérer la superposition entre deux objets *ch-FOF* superposés. Des arguments supplémentaires de fade-in et fade-out sont donnés aux objets *ch-FOF* pour créer un effet de staccato. A droite : schéma de la transformation d'un flux de trois FOF grâce à la fonction de fondu-enchaîné.

Le système que nous mettons en place ici permettra également de gérer de manière séquentielle les événements à traiter, à partir du mécanisme proposé par *ch-transitions*. Il s'agira de prendre en entrée des événements de contrôle, soit des instances des classes ch-FOF et ch-f0 et de produire les séquences d'états stables et de transitions correspondantes, à l'aide de profils formantiques donnés.

Il faudra alors être capable de coupler au système gestion des *profils formantiques*, un système de gestion des transitions de fréquence fondamentale. En effet, dans le contexte de la voix chantée, les phénomènes apparaissant lors de la transition entre deux notes ne dépendent pas que de l'évolution des formants, mais également de la f0.

Notre objectif est de proposer un système limitant le moins possible l'utilisateur en termes d'expressivité : d'une part, il doit être capable de définir ses propres transitions, sans limite concernant leur nombre ; d'autre part, bien que notre modèle de production vocale se base sur le chant lyrique, nous voulons être capable de dépasser ces limites et travailler avec des matériaux sonores ne faisant pas partie du « répertoire sonore » du bel canto.

FOF-transitions

C'est en se basant sur le système *ch-transitions*, décrit précédemment, qu'une première fonction, *FOF-transition*, a été implémentée. Cette fonction prend en entrée la définition du profil formantique désiré, ainsi que deux événements de contrôle du type *ch-FOF*, les transforme si besoin est, et insère entre les deux un nouvel événement de transition (également du type *ch-FOF*). Ainsi, utilisée comme fonction-lambda avec *ch-transitions*, elle permet de transformer une séquence d'événements ch-FOF.

La définition du profil formantique donné en paramètre se fait par l'intermédiaire de la classe

fmt-profile. Celle-ci est également une matrice permettant de stocker des courbes, correspondant respectivement aux trajets effectués par chacune des fréquences, des amplitudes et des largeurs de bande des formants (et extensible à d'autres paramètres). Est aussi stockée la courbe d'interpolation entre les deux états stables.

Voici un exemple de traitement d'une transition avec *FOF-transition*, sous forme schématisée :

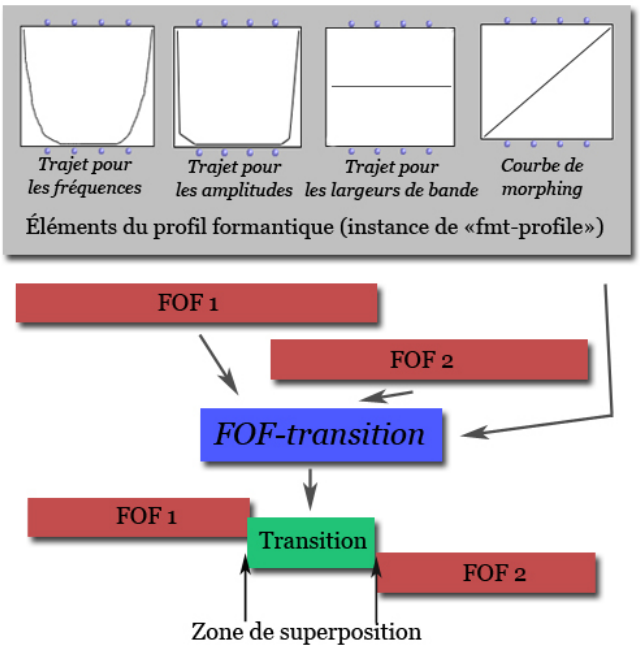

Figure 13 – Illustration du fonctionnement de *FOF-transition*, qui prend en entrée deux événements « FOF », et insère une transition à l'endroit où ils se superposent, conformément au profil formantique de la consonne 'b' (simplifié, tous les formants ayant le même profil) passé en paramètre.

La fonction présente deux modes de fonctionnement, que nous appellerons mode « implicite » et mode « explicite » :

- Dans le mode « implicite », la fonction ne se base que sur les propriétés temporelles des événements pour créer la transition. Deux cas peuvent se présenter : superposition (cf. figure 13) et événements distants : dans les deux cas, la longueur de la transition est définie respectivement par la durée de superposition et celle séparant les deux FOF.
- Dans le mode « explicite », une durée est donnée en tant que paramètre optionnel, qui fixe la longueur de la transition. Cette dernière sera placée avant le début du second état stable.

L'existence de ces deux modes est due à la volonté à de maximiser l'expressivité du système. En effet, bien qu'il soit possible de traiter un simple flux d'événements, il est aussi possible de donner en entrée le flux ainsi qu'une suite de durées de transitions, ce qui est judicieux si l'on veut effectuer, par exemple, une suite d'articulations de plus en plus courtes, via l'utilisation d'une suite de durées décroissante. Dans ce cas, on prend l'hypothèse que le flux d'événements donné en entrée n'est composé que d'objets se juxtaposant parfaitement dans le temps, l'événement de transition

prenant place pendant la fin de la première FOF [17].

Du point de vue algorithmique, *FOF-transition* effectue dans un premier temps le « morphing » (ou interpolation) entre les paramètres des deux FOF passées en entrée, puis échantillonne les trajets formantiques avant de les multiplier au résultat.

Ce processus pose le problème du taux d'échantillonnage : quelle granularité appliquer à l'événement de transition ? Nous avons alors introduit un paramètre optionnel permettant de régler le taux d'échantillonnage de la transition. Par défaut, la période d'échantillonnage est de 1ms. Cette durée à été validée expérimentalement, en testant des granularités différentes : il s'est avéré qu'une fréquence d'échantillonnage de 1000Hz convenait dans le cas des consonnes et des articulations communes type staccato et legato, sans poser de problèmes au niveau des performances (temps de calcul négligeable, de l'ordre de quelques millisecondes pour le rendu d'une seconde de trajets formantiques).

Une fonction analogue, *f0-transition*, permet de définir le trajet effectué par la fréquence fondamentale entre deux événements de type f0. Le trajet est formalisé par une courbe, qui vient multiplier l'interpolation linéaire effectuée entre les deux f0 qui jouxtent à la transition.

Dictionnaire

La gestion des différentes définitions de profils formantiques est centralisée, via l'utilisation d'un « dictionnaire ». Il s'agit d'un patch OpenMusic associant à une liste d'identifiants (clés [18]), représentés par des chaînes de caractères [19], une liste d'objets de type *fmt-profile*. Il devient alors possible de définir un dictionnaire de transitions, qui peut être étendu à volonté par l'utilisateur.

Le dictionnaire peut aussi avoir un rôle créatif, en dehors de celui de « base de données » : il est possible de faire la requête d'un profil formantique hybride, en sélectionnant deux clés correspondant à deux profils distincts, ainsi qu'un entier permettant d'effectuer une pondération entre ces deux derniers.

Transitions paramétrées

Nous avons voulu introduire un aspect « dynamique » dans notre système de transitions. En effet, nous sommes partis du cas pratique suivant : comment créer une phrase contenant des articulations évoluant progressivement du *legato* au *staccato*, sans avoir à modéliser chaque articulation intermédiaire entre les deux types de jeu ?

Nous avons alors mis en place un mécanisme permettant de paramétrer le profil formantique associé, en utilisant le système de fonctions d'ordre supérieur, pour permettre à *fmt-profile* d'être instancié dynamiquement pendant l'itération et en fonction de l'évolution des processus. Il devient alors possible d'imaginer une *transition* pouvant modéliser le staccato, le legato, ainsi que l'ensemble des profils intermédiaires entre ces deux articulations, grâce au système d'hybridation décrit précédemment. Un exemple d'application d'un tel type de transition est donné dans la partie 5.2.2.

17. Ici, la « FOF » signifie l'objet ch-FOF, par abus de langage.
18. « clés primaires », pour utiliser le vocabulaire des bases de données
19. Chaque clé ainsi définie est considérée comme unique.

4.3. Morphing formantique : une approche événementielle verticale

En parallèle à ce travail, nous avons implémenté une procédure permettant de générer des états hybrides entre stabilité et transition. Bien que cela n'entre pas directement dans le cadre de notre modélisation de départ, cette procédure permet d'étendre les possibilités compositionnelles offertes par le système.

Nous appellerons cette procédure le « morphing formantique ». En effet, elle permet de prendre en entrée plusieurs états stables, et de pondérer leurs paramètres, afin d'obtenir un état hybride. C'est selon ce principe que la fonction *FOF-morph* a été implémentée. Elle permet de prendre en entrée un ensemble d'événements FOF, ainsi qu'une courbe, et effectue une pondération dans le temps entre les événements qui se chevauchent, en suivant le profil donné par la courbe. Une courbe supplémentaire peut être définie afin de gérer différemment le morphing, en fonction du nombre d'événements se chevauchant : le premier profil sera utilisé si deux FOF se superposent, le second si une troisième FOF est donnée : on parlera alors de « morphing 3D ».

Sur la figure 14, on peut observer une illustration du mécanisme de morphing 3D :

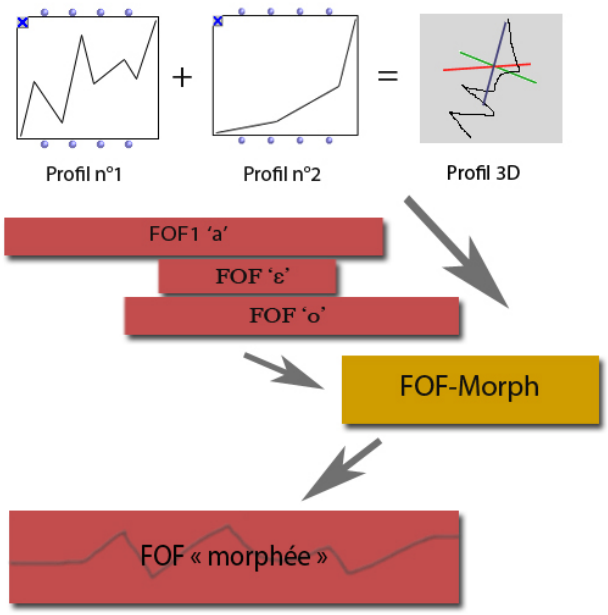

Figure 14 – Illustration du fonctionnement de *FOF-morph*, en effectuant une pondération entre trois états stables superposés, représentant les phonèmes 'ɜ', 'a' et 'o', en suivant deux profils (représentés en vue 3D).

5 Applications pratiques des procédures de contrôle

5.1. Introduction à l'opéra *Re Orso*

Lors des premiers mois de ce stage, nous avons eu l'occasion de collaborer avec le compositeur Marco Stroppa, en résidence à l'IRCAM pour travailler à la réalisation des sons électroniques nécessaires à la production de l'opéra *Re Orso*, qui fut créé le 19 mai 2012 à l'Opéra Comique de Paris [20].

Re Orso, pièce pour 4 chanteurs, 4 acteurs, ensemble, électronique, spatialisation et totem acoustique, conte les dernières heures du Roi Ours, qui régnait sur la Crète avant l'an 1000, et qui fut tristement célèbre pour sa cruauté et ses excès. Au bord de l'agonie, il fait la rencontre du Ver, entité mystérieuse, qui le torture en le confrontant à son comportement inique.

Le travail que nous avons effectué avec Marco Stroppa avait pour but de synthétiser des sons de voix sensés être produits par le Ver, inspirés d'un modèle de voix imaginaire. En effet, le Ver est un être immatériel, non humain, représenté sur scène de manière symbolique, mais en réalité présent dans la tête du Roi Ours. C'est pourquoi les voix produites par le Ver se devaient de posséder un caractère « non-humain », sans être des voix que l'on pourrait qualifier de « voix d'ordinateur ». Le choix de Marco Stroppa a donc été de produire des voix dépassant, « pervertissant » les capacités de l'appareil vocal d'un chanteur. [21]

5.2. Exemples d'applications

Nous allons maintenant présenter plusieurs travaux compositionnels, effectués avec Marco Stroppa, dont le rendu sonore a été utilisé lors de la création de *Re Orso*.

5.2.1 Création de phrases vocales simples

Les premières applications du système ont consisté en la création de phrases vocales simples. Elles ont permis de tester la souplesse du système, dans un contexte de mélange voyelles/consonnes basique.

Nous avons tout d'abord modélisé des sortes de babillements, en alternant voyelles, et consonnes faciles à modéliser (en l'occurrence, le 'b' et le 'm'). D'un point de vue musical, des résultats intéressants ont été obtenus en faisant varier la durée des consonnes au sein d'une même phrase.

Un cas particulier a été étudié : celui du roulement de 'r'.
Contrairement au 'ʀ' tel qu'on le prononce en français [22], qui peut être synthétisé à l'aide d'un *jitter*

20. Informations supplémentaires disponibles sur http ://www.opera-comique.com/spectacle/re-orso.

21. Cette démarche a été exploitée dans les années 1980 par Jean-Baptiste Barrière, à l'occasion de la production de la pièce « Chréode » [Barrière, 1984]. Selon Barrière, il était question de « repousser les limites [de la voix chantée] sans que ça semble incohérent au niveau perceptif ».

22. Il s'agit d'une consonne dite *uvulaire* en phonétique.

(une variation aléatoire) agissant sur les différents paramètres formantiques, le ‘r’ roulé présente un motif périodique. En effet, en fonction de la morphologie de l’appareil vocal du chanteur, la langue vient frapper le palais à une certaine fréquence [23]. La figure 15 propose une analyse temps/fréquence, d’un ‘r’ roulé, d’une fréquence de 21Hz :

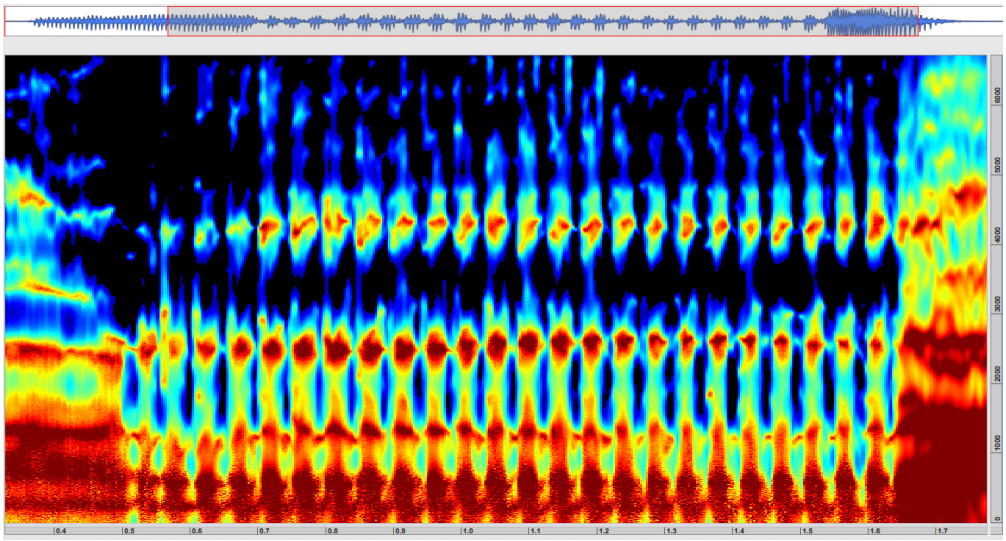

Figure 15 – Analyse de la syllabe ‘ara’, dont le ‘r’ comprend 24 claquements de langue, effectués à une fréquence de 21Hz.

Au regard de cette particularité, nous avons choisi de modéliser le ‘r’ roulé par une suite de *transitions*, chacune correspondant à une période, soit à un claquement de langue. Un ensemble d’états stables courts permettra de séparer les différentes transitions. Ainsi, il est possible de paramétrer le roulement selon deux variables : la durée de la transition (durée du coup de langue), et la durée des états stables situés entre les trajets. Cette modélisation, qui est bien entendue destinée à traiter un cas particulier, n’est pas très simple à utiliser dans la pratique : ce qui nous poussera plus tard à proposer un mécanisme pour gérer les suites de consonnes, et plus généralement, les suites de transitions.

Cependant, cette modélisation comporte certains avantages, qui furent mis à profit dans le cadre de *Re Orso* : il devient aisé de paramétrer la durée de chaque claquement, et donc d’effectuer des accelerando ou des rallentendo au niveau des roulements, chose impossible à effectuer pour un chanteur. En particulier, nous avons synthétisé le chant du mot « Re », présentant un roulement à fréquence variable, une fois de plus dans l’objectif de créer des voix dépassant les capacités de l’appareil vocal. Un exemple d’analyse d’un tel son est visible sur la figure 16.

23. La moyenne se situe approximativement autour de 20Hz, cependant, il existe des individus capables de rouler les ‘r’ à des fréquences dépassant les 30 Hz.

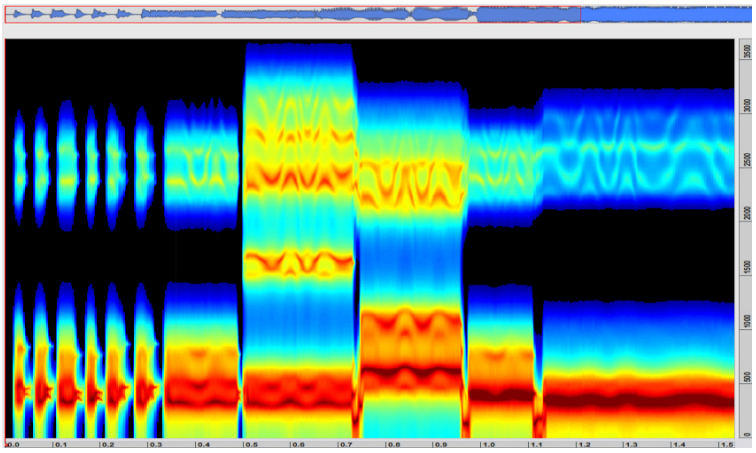

Figure 16 – Analyse d'un roulement de 'r' dont la fréquence décroît au cours du temps, synthétisé avec CHANT.

5.2.2 Evolution legato-staccato

Nous avons voulu ici générer une phrase de chant dans laquelle les articulations évoluent de manière « continue » du staccato au legato. Nous avons pour cela utilisé le mécanisme de *transitions paramétrées*, que nous avions introduit à la fin de la partie 4.2.

Le principe est le suivant : on définit une courbe, évoluant de 0 à 1, les deux nombres représentant respectivement l'aspect staccato et legato des articulations. Cette courbe est ensuite échantillonnée, et ses valeurs seront transmises comme paramètres à une transition, permettant d'obtenir les articulations intermédiaires entre staccato et legato. Des trajets formantiques d'amplitude, caractéristiques de ce type de transition paramétrée sont disponibles sur la figure 17 : le staccato est modélisé par un trajet en forme de 'u' dont le minimum est nul (silence). Plus on évolue vers le legato, plus ce minimum va tendre vers 1 (pas d'atténuation) et plus les pentes du trajet tendre vers zéro.

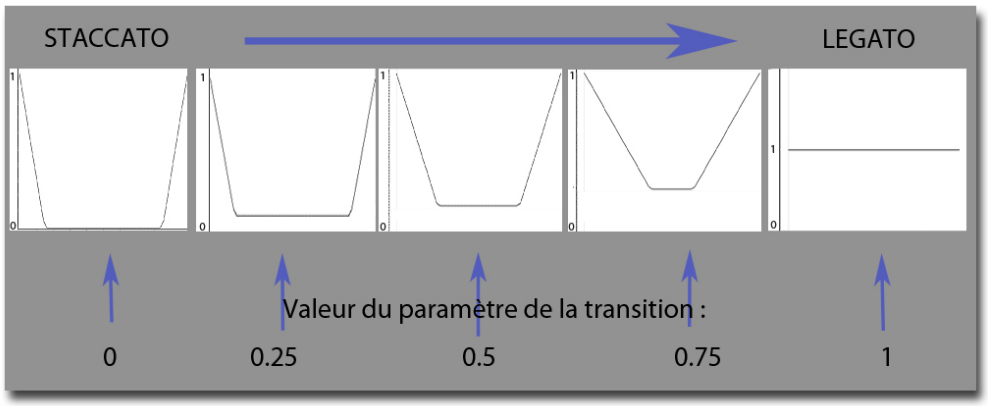

Figure 17 – Trajet formantique de l'amplitude du premier formant, selon différentes valeurs données en paramètre.

L'introduction du mécanisme de *transitions paramétrées* permet ici de créer une évolution perçue comme continue entre deux techniques de chant [24].

5.2.3 Transformation continue d'un matériau sonore non-organique en voix

Cet exemple permet de décrire un cas de synthèse par FOF dépassant le cadre de la voix chantée. En effet, nous avons ici voulu transformer progressivement une sonnerie de téléphone (sonnerie monophonique et mono-timbrale) en une voix de chanteur lyrique.

Pour cela, il est possible de travailler à partir de plusieurs éléments :

Paramètres	Téléphone	Voix
Paramètres des états stables	formants reproduisant le timbre du téléphone	formants issus de l'analyse d'un extrait d'opéra [25]
Articulations staccato-legato	staccato pur	legato ou staccato, selon l'effet désiré
Durée des transitions	< 25 ms	> 50 ms
Vibrato suivant une enveloppe sur chaque note	nul (f0 constante)	vibrato à forte amplitude

Les articulations ont été effectuées grâce à la transition paramétrée legato-staccato (cf. section 5.2.2), et les vibratos ont été générés sur chaque note grâce à une fonction écrite spécialement à cet effet.

Voici l'analyse spectrale du rendu sonore d'un tel processus, sur laquelle on peut observer l'évolution de ces paramètres :

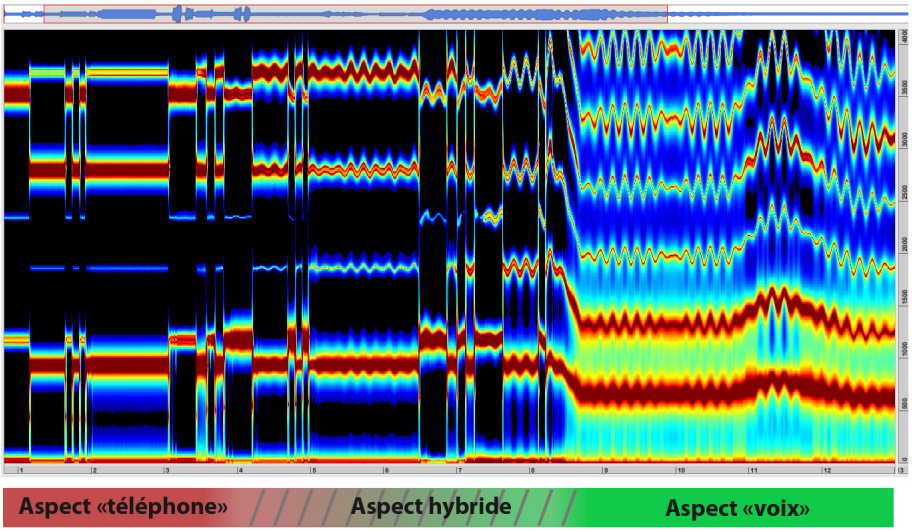

Figure 18 – Analyse spectrale du rendu sonore du processus « téléphone vers voix », indentification des trois « aspects » sonores.

24. Extrait sonore disponible sur http ://repmus.ircam.fr/cao/om-chant/examples.
25. Extrait de « La Reine de la Nuit », Mozart, interprétation de Erika Miklosa.

Ici aussi, nous avons réussi à obtenir un rendu sonore qui donne l'impression d'une évolution continue entre deux processus de synthèse différents, à la manière d'un *morphing*[26]. Nous pouvons entendre ce son lors des premières secondes de *Re Orso*.

5.2.4 Limites de la première version du système de contrôle

Bien que notre système permette de larges possibilités créatives, il souffre de quelques lacunes. Par exemple, la définition des événements à traiter se fait de manière trop « statique » : selon les objets donnés en entrée au système, on n'est pas toujours capable de modifier de manière efficace (au niveau du temps de travail) la structure temporelle des états stables qui vont former le corps de la phrase. S'ils sont générés à partir d'une représentation de partition du type *chord-seq*[27], il est aisé de déplacer un groupe de notes dans le temps, mais s'ils sont créés à partir d'une liste d'onsets et de durées, le travail devient beaucoup plus long.

Il devient alors nécessaire de réfléchir à la conception d'un mécanisme qui permettrait de modifier de manière plus intuitive et efficace les structures musicales utilisées.

26. Un extrait sonore est disponible sur http ://repmus.ircam.fr/cao/om-chant/examples.
27. Dans OpenMusic, les instance de la classe chord-seq permettent de représenter des suites de notes ou d'accords.

6 Environnements graphiques et contrôle de haut niveau

Dans l'environnement OpenMusic, nous allons utiliser le mécanisme de *maquettes* afin proposer une seconde version du système, permettant de pallier aux lacunes du premier (cf. section 5.2.4).

6.1. Implémentation de procédures de contrôle au sein des maquettes OpenMusic

6.1.1 Maquettes OpenMusic

Dans l'environnement OpenMusic, les maquettes sont des interfaces permettant de représenter et d'agencer des structures musicales dans le temps. Elles sont composées d'une interface principale, comprenant une « timeline », sur laquelle il est possible de déposer ou de créer des patches ; ainsi que d'un patch interne d'évaluation [Bresson et Agon, 2006]. Ce dernier permet de définir des programmes visuels, implémentant le comportement de la maquette, c'est à dire définir le processus produisant le résultat de la maquette (rendu sonore ou autre). Une maquette peut, comme tout patch OpenMusic, fournir en sortie n'importe quel type de donnée. Cependant, si l'évaluation de la maquette donne lieu à un rendu sonore, un player intégré permet de jouer ce dernier, en suivant sa progression grâce à une tête de lecture se déplaçant dans l'interface.

Les objets manipulés au sein des maquettes sont appelés *temporalbox*. Il s'agit de « super-patches » contenant des informations, d'ordre temporel (date de début, durée) ou autre, telles que leurs dimensions, leur position et leur couleur, et qui peuvent être évaluées individuellement. Ces objets, ou modules, peuvent être reliés entre eux comme éléments d'un programme visuel, à l'aide d'entrées ou de sorties. Il est aussi possible de librement modifier leur position dans la maquette, ainsi que leurs dimensions.

Deux idées ont motivé l'utilisation des maquettes au cours de ce stage : la représentation symbolique des différents éléments de contrôle sur un espace 2D, et la possibilité d'éditer de manière intuitive et efficace les phrases musicales.

6.1.2 Adaptation et ré-organisation du système

Le principe de l'intégration des procédures de contrôle au sein des maquettes suit la même logique événementielle que précédemment : des objets représentant des FOF ainsi que des valeurs de f0 pourront être agencées dans le temps, et interagir pour produire de nouveaux événements, tels que des transitions ou des morphings. Nous appellerons « modules » ces objets, de type *temporalbox*, dont la valeur [28] est une instance des classes *ch-fof* ou *ch-f0*. Nous avons dans un premier temps implémenté plusieurs fonctions permettant un contrôle basique de CHANT :

28. La valeur d'un module est le résultat de son évaluation.

– *maq-FOF*, qui génère un événement de type ch-FOF, prenant en entrée des paramètres formantiques.
– *maq-f0*, et ses variantes, *maq-f0-bpf*, *maq-f0-vib* et *maq-f0-jit*, qui permettent respectivement de générer un événement de f0 constante, dépendant d'une courbe, avec un vibrato ou avec jitter [29]. L'environnement interactif proposé par les maquettes est ici mis à profit, puisqu'il est possible de régler plusieurs paramètres en fonction des propriétés spatiales des modules : leur position sur l'axe vertical définit leur fréquence centrale, et leur largeur permet de régler l'amplitude maximale d'un vibrato ou d'un jitter.

A ces premières fonctions s'ajoutent *maq-FOF-transition*, *maq-FOF-morph* et *maq-f0-transition*, qui permettent, comme leur noms l'indiquent, de produire des modules de transitions ou des morphings. Ces derniers comportent chacun deux entrées, qui pourront être connectées à d'autres modules présents dans la maquette, comme l'illustre la figure 19 :

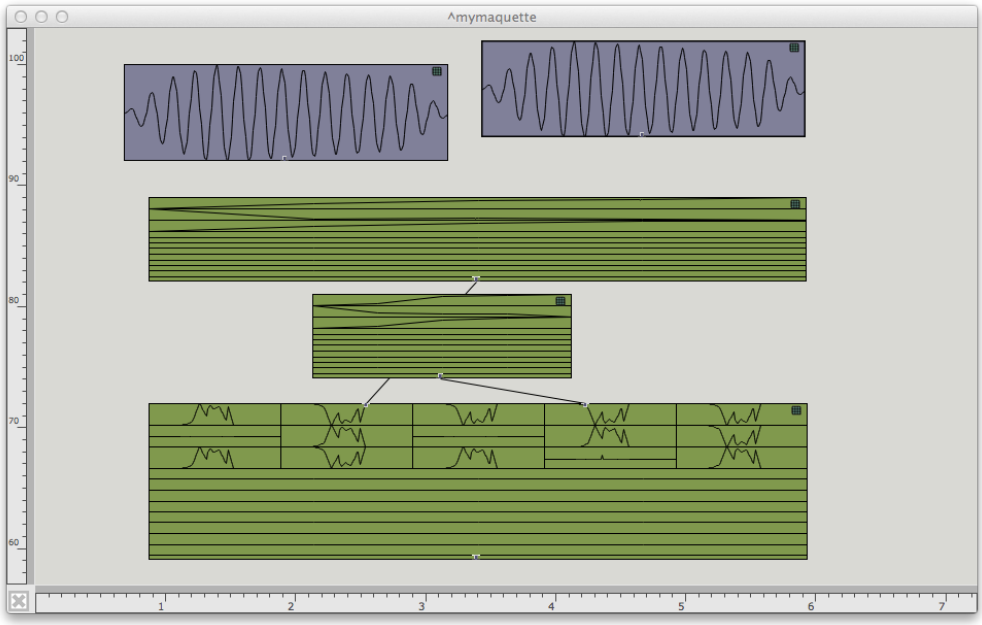

Figure 19 – Maquette contenant deux modules de f0 avec vibrato (en haut), ainsi que deux modules FOF (au milieu), connectés puis transformés en un module de morphing (en bas).

L'ensemble des objets produits par création de transition ou par morphing sont de type homogène : par exemple, le résultat du morphing entre deux modules FOF, générant chacun des valeurs de type ch-FOF, sera de type ch-FOF (il en est de même pour les modules de f0). Cela permet de dépasser l'approche purement séquentielle de la gestion des morphings et des transitions telle qu'elle est effectuée dans la première version du système : il est ici possible de définir des morphings de morphings (ce qui permet plus de possibilités que le morphing 3D), des morphings de transitions, et bien entendu des transitions entre morphings. A noter que lorsque deux modules de FOF sont « morphés » en un autre, ils ne sont plus pris en compte dans la séquence de contrôle du synthétiseur, où ils sont substitués par le résultat du morphing.

29. Des enveloppes d'amplitude et de fréquence peuvent être définies pour le vibrato/jitter.

6.2. Définition d'une phrase vocale à partir de structures de haut niveau

Les maquettes OpenMusic peuvent être utilisées comme interface de conception de programmes et de séquences temporelles, mais aussi être générées algorithmiquement au sein d'autres programmes visuels.
Après avoir défini les différents modules représentant les événements de contrôle, nous avons implémenté plusieurs fonctions permettant de remplir automatiquement la maquette avec des modules de FOF et de f0, correspondant à une phrase définie à partir de structures musicales de haut niveau (suites de notes, listes de durées, de rythmes...).

6.2.1 D'une syntaxe abstraite à la représentation graphique d'une phrase

Deux fonctions ont été conçues dans le but de générer une maquette, avec des modules de FOF pour la première et de f0 pour la seconde.

La première, *make-temporal-FOF*, prend en entrée :

- une structure temps-fréquence [30], représentant différentes notes jouées (on peut par exemple partir d'un *chord-seq*).
- une suite de symboles, désignant les clés pointant vers des modules d'états stables et de transitions.
- une base de données effectuant la relation clés/modules de FOF (instance de la classe *phoneme-DB*).
- une liste contenant les durées des différentes transitions.

La fonction *make-temporal-FOF* analyse ces entrées, vérifie qu'elles ne sont pas incohérentes entre elles (pas assez d'états stables par rapport au nombre de notes, transitons trop longues par rapport aux états stables...), puis génère les modules qui viendront remplir la maquette cible. Bien que dans la plupart des cas la fonction ne fasse que créer et câbler les modules idoines, elle effectue également un traitement particulier aux suites de transitions ne comprenant pas d'états stables à l'intérieur, comme dans une suite de consonnes. En effet, une suite de transition n'est pas en accord avec notre modèle de production vocale, qui est basé sur la succession état-stable/transition [31]. *Make-temporal-FOF* insère alors entre chacune des transitions un état pseudo-stable, d'une durée donnée, de mêmes paramètres formantiques que l'état stable précédant le groupe de transitions, ce qui permet de maintenir l'alternance état stable/transition. Une variante de cette procédure pourrait consister à reproduire les formants correspondant au phonème 'ɜ', qui est souvent trouvé entre des suites de consonnes ('bbb', 'mmm').

La classe *phoneme-DB*, reprend l'idée de « dictionnaire », utilisée dans le système présenté dans la section 4. Elle permet de classer des références à des *temporalbox*, créées à partir de fonctions

30. Structure sous la forme d'une liste d'éléments (fréquence, date, durée).
31. Rappelons qu'une transition ne fait que décrire l'évolution temporelle de paramètres formantiques.

telles que *maq-FOF* et *maq-FOF-transition*, qui iront remplir une maquette.

Grâce à ce système de base de données, nous pouvons par exemple concevoir des « banques de chanteurs », qui contiennent les phonèmes et les articulations propres aux formants et au style de jeu d'un chanteur virtuel.

Une seconde fonction de « remplissage », *make-temporal-f0*, permet de générer des modules de f0. Elle prend elle aussi en entrée une structure temps-fréquence et une suite de durées de transitions, ainsi que deux références [32], l'une au module de f0 souhaité (f0 constante, avec vibrato, etc...), et l'autre à module de transition de f0 souhaité. Ainsi, l'utilisateur peut définir un profil de transition entre modules de f0 qui sera appliqué entre chaque événement de f0 distant. Cela permet un contrôle plus riche qu'en effectuant une simple interpolation entre deux fréquences, ce que fait par défaut le système CHANT, et permet de reproduire les oscillations effectuées entre deux notes par la voix, autour d'une fréquence centrale [Sundberg, 1987]. Un exemple de transition entre deux modules de f0 est présenté sur la figure 20.

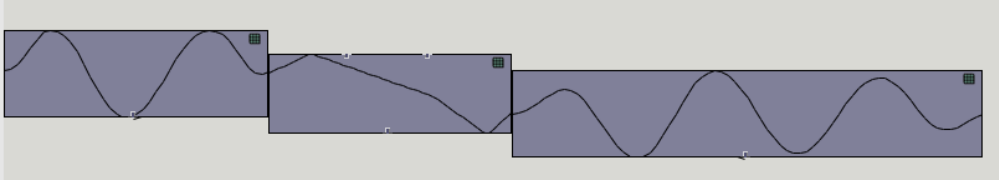

Figure 20 – Module de transition de fréquence fondamentale, connecté entre deux modules de f0. Cette transition reproduit le trajet oscillant effectué par la f0 durant la transition.

Les deux fonctions de génération de maquette peuvent être réunies au sein de *make-temporal-FOF-f0*, qui prend en entrée les mêmes arguments que ceux de *make-temporal-FOF* et de *make-temporal-f0* réunies. La figure 21 illustre le remplissage d'une maquette via l'utilisation de structures musicales de haut niveau :

32. On parle ici de références à des patches OpenMusic.

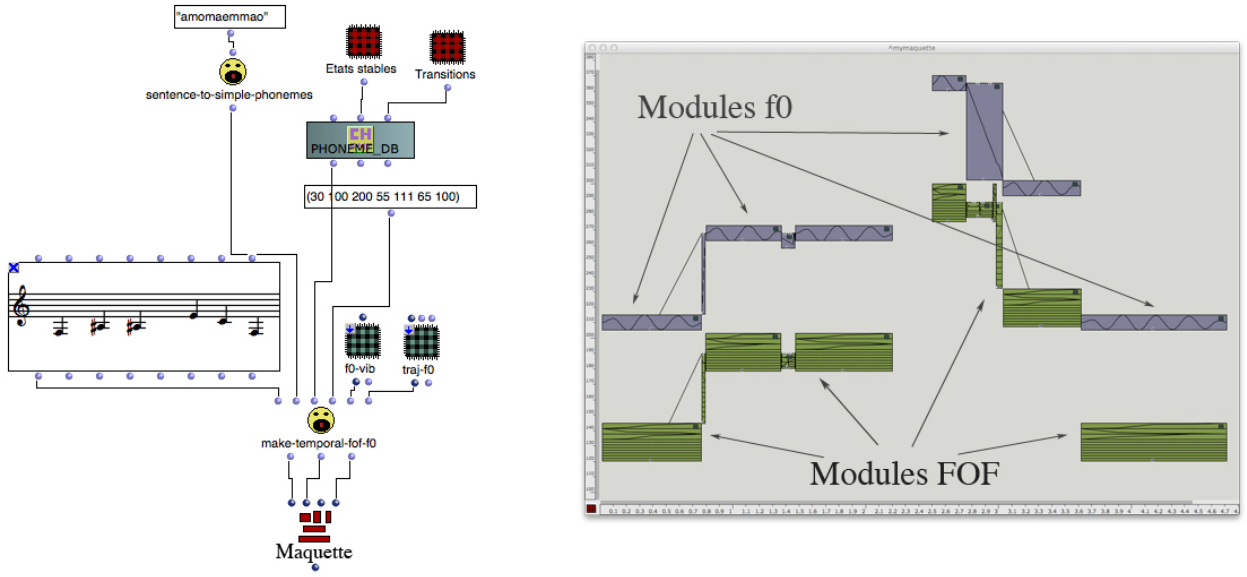

Figure 21 – Génération d'une maquette à l'aide de la fonction *make-temporal-FOF-f0*. Les données initiales sont un *chord-seq*, une chaîne de caractères, une base de données de type *phonème-DB*, une suite de durées et les références aux modules de f0. A gauche : vue du patch principal. A droite : résultat dans l'éditeur de maquette.

6.2.2 Gain en expressivité du nouveau système

Cette nouvelle version du système permet de travailler à deux niveaux : à partir de structures musicales de haut niveau (chord-seq, séquences de « phonèmes »), et directement à partir de la maquette.

Le premier a deux vocations : le travail à partir de données musicales précises (si l'on a, par exemple, les durées exactes des notes et des transitions), et le remplissage automatique des maquettes, afin de gagner du temps en générant une ébauche de phrase musicale.

Le second niveau permet quant à lui d'agencer les événements de contrôle dans le temps (et pour certains, de régler des paramètres de synthèse en modifiant leurs positon et dimensions), et ce de manière intuitive, grâce à l'interface graphique.

Ce contrôle à deux niveaux, ainsi que les nouvelles possibilités de morphing [33], permettent donc à l'utilisateur une plus grande liberté compositionnelle, et fait gagner en expressivité les procédures de transition et de morphing implémentées plus tôt.

Nous effectuerons d'ailleurs, dans la section suivante, un retour sur le travail réalisé pour l'opéra *Re Orso*, afin d'illustrer les améliorations fournies par le nouveau système.

6.2.3 Limitations

Contrairement au système initial, dans lequel il est possible de créer transitons et morphings de manière implicite, par simple chevauchement ou insertion d'un écart entre deux événements, il

33. morphing de morphings / de transitions

n'est pas ici possible de définir une transition (ou un morphing) sans la création d'un objet idoine.

Ainsi, les deux systèmes que nous avons élaborés ont chacun leurs avantages et inconvénients, mais permettent de travailler selon un *workflow* qui leur est propre.

6.3. Un retour vers *Re Orso*

Nous avons utilisé ce nouveau mécanisme basé sur les maquettes afin de créer des « classes de patches », des « templates », reproduisant des processus pensés pour *Re Orso*. En effet, grâce aux améliorations apportées par cette nouvelle version du système, il devient possible de produire toute une famille de processus musicaux dérivant d'une même pensée compositionnelle originale.

En particulier, nous avons reproduit le processus musical qui nous a de loin demandé le plus d'efforts lors de la création de la partie électronique de *Re Orso* : le glissando de la mort du roi. Cet effet vise à transformer, sur une durée de trois minutes, la voix du Roi Ours [34], en un glas, modélisé à partir de l'analyse temps-fréquence d'un enregistrement des cloches de l'abbaye de Westminster.

Le glissando en lui-même se déroule ainsi : on part de la voix du Roi Ours, qui passe par plusieurs états rapides, modélisés par des paramètres formantiques correspondant à la voyelle 'u', en passant d'un timbre vocal de soprano à alto, puis à basse. Puis, on assiste à une phase de transition, lors de laquelle la f0 descend progressivement, en effectuant par moments quelques vibratos. Enfin, lors du glas final, on se rapproche de plus en plus du timbre de la cloche alors que la f0 tend vers 0. Un paramètre supplémentaire que nous n'avions pas évoqué précédemment est ici modifié : il s'agit de la durée de la sinusoïde amortie, qui est ici étendue à 5 secondes afin d'obtenir la résonance nécessaire à la synthèse des sons de cloches. Cependant, cette modification reste anecdotique au niveau du contrôle, car il suffit d'activer un argument optionnel lors de la création du dernier module de FOF.

Un très grand nombre de FOF est généré à chaque impulsion du train (171 au total), afin de reproduire le plus de partiels possibles appartenant au spectre de la cloche. De plus, la largeur de bande de chacune de ces FOF tend vers 0 au fur et à mesure que l'on se rapproche du son de cloche. Nous ne faisons alors plus de synthèse « par formants », mais tentons de recréer un processus de synthèse additive, en effectuant une addition de sons purs (leur largeur de bande tendant vers 0) à différentes amplitudes.

Voici ce que donne la réalisation finale, simplifiée et réduite à une durée de 30 secondes, sous forme de maquette OpenMusic commentée :

34. du moins, de son interprète, modélisée directement à partir de ses formants

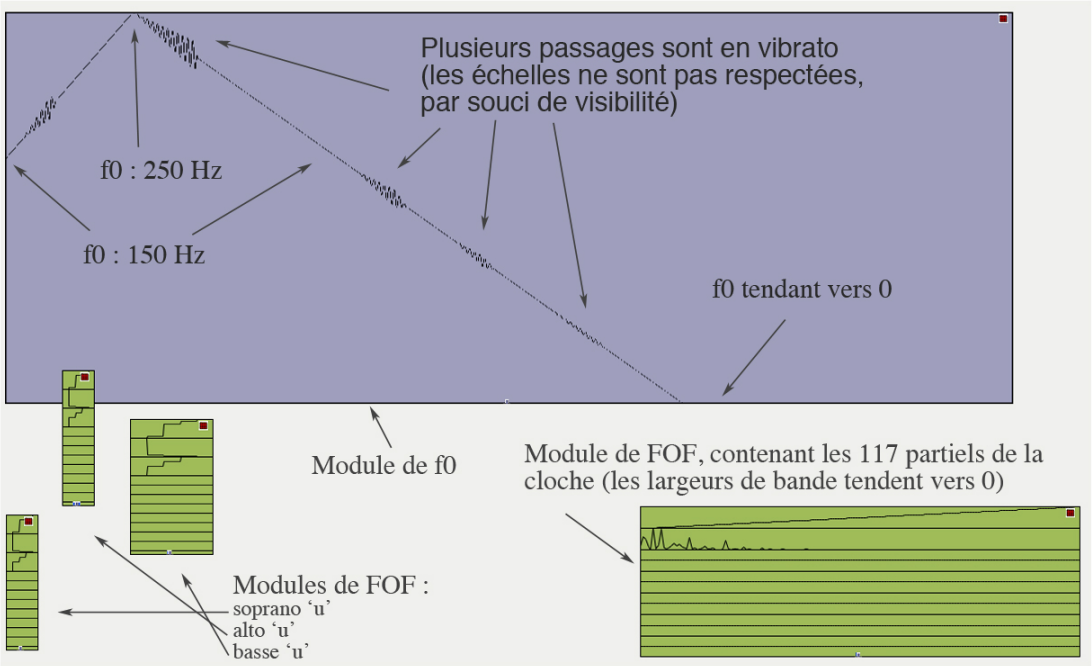

Figure 22 – Maquette du processus de glissando final, avec sa représentations symbolique.

6.4. Intégration des procédures de contrôle à la bibliothèque *OM-Chant*

L'ensemble des outils dont nous avons discuté au cours de ce rapport ont été intégrés à la bibliothèque OM-Chant, dans sa version 2.0 (cette version sera distribuée dans l'offre Forum Recherche de l'IRCAM en novembre 2012) . Les procédures de contrôle ainsi ajoutées à OpenMusic peuvent se diviser en trois classes d'utilisation :

- Procédures de contrôle « séquentiel » de la synthèse : traitement d'un flux d'événements au moyen de listes de données temporelles et fréquentielles.
- Outils pour travailler au sein des maquettes OpenMusic : création, interaction intuitive avec les boîtes, et gestion des événements par câblage interne.
- Mécanismes de remplissage automatique de maquettes, à partir de structures musicales symboliques, et de bases de données prédéfinies ou créées par l'utilisateur.

De plus, un ensemble de tutoriels proposent une prise en main du système via des exemples originaux[35] :

- Création d'une ligne de voix simple, à partir d'un objet chord-seq, d'une chaîne de caractères et d'une base de données de phonèmes.

35. Ces exemples sont disponibles sur http ://repmus.ircam.fr/cao/continuous-control.

– Utilisation de transitions paramétrées dans le cadre du contrôle continu de l'aspect staccato/legato d'un ensemble d'articulations.
– Processus de transformation d'un matériau non organique en voix, en mode maquette (simplification du patch « téléphone vers voix » conçu pour *Re Orso*).
– Création d'un duo virtuel, jouant en contrepoint, en utilisant des fonctions de remplissage automatique de maquettes.

7 Conclusion et perspectives

Nous avons introduit dans l'environnement OpenMusic plusieurs méthodes permettant d'élaborer des procédures de contrôle originales du synthétiseur CHANT. En partant d'un modèle de base pour la voix chantée que nous avons étendu afin de maximiser les capacités expressives du système. Deux mécanismes ont été mis en place : un premier permettant de gérer une séquence d'événements en les traitant deux à deux, et un second permettant de manipuler les événements de synthèse de manière intuitive grâce à une interface graphique.

Les deux versions du système que nous avons implémentées ont été intégrées à la bibliothèque OM-Chant, dont la prochaine version sera mise à disposition du public courant 2012. Plusieurs tutoriaux l'accompagneront, ainsi que quelques ressources proposant une base de données de profils formantiques, permettant une rapide prise en main.

La présence de Marco Stroppa nous a permis d'obtenir un recul très instructif sur notre travail, en confrontant notre vision du système avec son point de vue de compositeur. De plus, travailler sur une production telle que *Re Orso* fut une expérience très motivante, et très gratifiante une fois la pièce créée.

Notre collaboration n'ayant eu lieu que durant le début de ce stage, seule la première version de notre système – n'intégrant pas les maquettes – a été exploitée lors de cette collaboration. Cependant, les nouvelles procédures que nous avons conçues pourront être utilisées lors du raffinement de certains processus de synthèse, qui seront mis en place à l'occasion de la prochaine « version » de l'opéra *Re Orso*, dont plusieurs représentations sont prévues dans les années à venir.

Afin de terminer notre modèle de production vocale, il faudrait effectuer un travail en profondeur sur l'ajout de composantes bruitées. Ces fonctionnalités n'ont pas été proposées, afin de nous concentrer sur l'étude des sons voisés, qui constitue à elle seule un sujet de recherche très vaste. Cependant, une fois les bons paramètres trouvés, les procédures proposées par notre système pourront rapidement intégrer le contrôle de la synthèse de sons non-voisés.

La synthèse par FOF est dotée d'un potentiel créatif très large, notamment de par la richesse des gammes de sons qu'elle est capable de produire. Si nous nous sommes souvent limités à la production de signaux inspirés de la voix chantée, des procédures de contrôle permettant de reproduire (et de pervertir !) d'autres types de sons pourront à l'avenir être imaginées. Parmi eux, certains sons instrumentaux qui se prêtent bien à la synthèse par formants (percussions, cloches, cuivres...), ou encore d'autres sons issus de techniques originales, telles que la synthèse granulaire [Liuni et Gentilucci 2010].

8 Bibliographie

[Agon 1998] C. Agon, « *OpenMusic : Un langage visuel pour la composition musicale assistée par ordinateur* », PhD Thesis, IRCAM-Paris VI, Paris.

[Agon et al., 2011] C. Agon, J. Bresson, M. Stroppa., « OMChroma : Compositional Control of Sound Synthesis », *Computer Music Journal*, The MIT Press, 35(2), 2011.

[Anderson et Kuivila, 1989] D. P. Anderson et R. Kuivila, « Continuous Abstractions for Discrete Events Languages », *Computer Music Journal*, The MIT Press, 13(3), 1989.

[Assayag 1998] G. Assayag, « Computer Assisted Composition today », *1st symposium on music and computers*, Corfu, Grèce, 1998.

[Assayag et al., 1999] G. Assayag, C. Agon, M. Laurson, C. Rueda : « Computer Assisted Composition at Ircam : Patchwork and OpenMusic », *Computer Music Journal*, The MIT Press, 23(3), 1999.

[Baisnee, 1985] P-F. Baisnee and the Chant Group, « Chant Manual », IRCAM, fev. 1985.

[Barrière, 1984] J-B. Barrière, « Chréode, un chemin vers une nouvelle musique avec l'ordinateur », *IRCAM : une pensée musicale*, InterEditions, 1984.

[Bogaards et Röbel, 2004] N. Bogaards et A. Röbel, « An interface for analysis- driven sound processing », *AES 119th Convention* , New York, USA, 2004.

[Bresson et Agon, 2006] J. Bresson et C. Agon, « Temporal Control over Sound Synthesis Processes », *Proc. Sound and Music Computing Conference*, Marseille, 2006.

[Bresson et Agon, 2007] J. Bresson et C. Agon., « Musical Representation of Sound in Computer- Aided Composition : A Visual Programming Framework », *Journal of New Music Research*, 36(4), 2007.

[Bresson et al., 2009] « Visual Lisp/CLOS programming in OpenMusic », *Higher-order and symbolic computation*, 22(1), 2009.

[Bresson et Michon, 2011] J. Bresson et Michon, « Implémentations et contrôle du synthétiseur CHANT dans OpenMusic », *Actes des Journées d'Informatique Musicale*, Saint-Etienne, France, 2011.

[Bresson et Stroppa, 2011] J. Bresson et M. Stroppa, « The Control of the CHANT Synthesizer in OpenMusic : Modelling Continuous Aspects in Sound Synthesis », *Proc. Int. Computer Music Conference*, Huddersfield, Royaume Uni, 2011.

[Li et al, 2003] D. Li et D. O'Shaughnessy, « Speech processing : a dynamic and optimization-oriented approach », Marcel Dekker Inc., New York, 2003.

[Liuni et Gentilucci 2010] M. Liuni et M. Gentilucci, « Multifog : a Multi-Level Control Device for FOG Granular Synthesis in Max/MSP », *Computer Music Modeling and Retrivial*,

Malaga, Espagne, 2010.

[Rodet et Cointe, 1984] X. Rodet et P. Cointe, « Formes : Composition and Scheduling of Processes », *Computer Music Journal*, The MIT Press, 8(3), 1984.

[Rodet et Depalle, 1985] X. Rodet et P. Depalle, « High quality synthesis-by-rule of consonants », *Proc. International Computer Music Conference*, Vancouver, Canada, 1985.

[Rodet et Lefevre, 1997] X. Rodet et A. Lefevre, « The Diphone Program : New Features, New synthesis Methods and Experience of Musical Use », *Proc. Int. Computer Music Conference*, Thessaloniki, Grèce, 1997.

[Rodet et al., 1984] X. Rodet, Y. Potard et J-B. Barriere, « The CHANT Project : From the Synthesis of the Singing Voice to Synthesis in General », *Computer Music Journal*, The MIT Press, 8(3), 1984.

[Rodet, 1984] X. Rodet, « Time-domain Formant-wave Function Synthesis », *Computer Music Journal*, The MIT Press, 8(3), 1984.

[Rodet, 1977] X. Rodet, « *Analyse du signal vocal dans sa représentation amplitude-temps : synthèse de la parole par règles* », PhD Thesis, Université Paris VI, 1977.

[Wright et al., 1999] M. Wright, A. Chaudhary, A. Freed, S. Khoury, D. Wessel, « Audio Applications of the Sound Description Interchange Format Standard », *Audio Engineering Society Convention 107*, New York, USA, 1999.

[Sundberg, 1987] J. Sundberg, « *The Science of the Singing Voice* », Northern Illinois University Press, Dekalb, IL, 1987.

[Sundberg, 2006] J. Sundberg, « The KTH synthesis of singing », *Advances in Cognitive Psychology*, Versita, 2006.