

UNIVERSITE PARIS 8 VINCENNES À ST-DENIS
UFR ARTS. PHILOSOPHIE, ESTHÉTIQUE
Département de Musique

Un guide sur l'analyse spectrale

Du vocodeur de phase au morphing

Emmanouil-Nikolaos KARYSTINAIOS

Mémoire de Master 2 réalisé sous la direction de M. Alain BONARDI

Année universitaire 2017-2018.

Resumé

Cette recherche se focalise, au fond, sur l'exploration de la nature spectrale du son, et plus précisément, le morphing spectrale en temps réel. Un point de vue pédagogique et documentaire est appliqué pour approfondir l'exploitation technique et musicale du spectre. En essayant d'expliquer la transformation de Fourier sur l'analyse spectrale sonore, physiquement et numériquement, en apposant les filtres Gabor pour le fenêtrage du son, en analysant le fonctionnement du Vocoder de phase, en recherchant la connexion entre le morphing sonore et visuel, nous retrouvons les détails qui nous permettront de découvrir des nouvelles manières pour la manipulation et la compréhension de ces notions.

La recherche, pour des raisons logiques et structurelles est divisée en trois parties. La première exploitera la théorie de tous les notions référencées ci-dessus. La réalisation des modèles théoriques est faite à partir du logiciel MaxMSP en décomposant toutes les étapes avant d'arriver au morphing Spectrale, dans la deuxième partie. La dernière partie de cette recherche, se concentre sur la proposition des applications artistiques du morphing et des autres effets spectraux qu'on a créé dans la partie précédente.

Table des matières

Abstract	i
1 Introduction	1
1.1 À propos	1
1.2 Structure	2
1.3 MaxMSP et Jitter	3
1.4 Analyse spectrale	4
1.5 Morphing	4
1.5.1 Morphing sonore	5
1.5.2 Morphing visuel	5
2 Context théorique	6
2.1 Introduction	6
2.2 Le décodage mathématique	7
2.2.1 La transformation Fourier	7
2.2.2 Fenêtrage et filtres Gabor	11
2.2.3 Le vocodeur de phase	13

2.2.4	Applications du vocodeur de phase	16
3	Design	20
3.1	MaxMSP	20
3.2	Pitch tracking	20
3.3	Analyse spectrale	23
3.4	Le vocodeur de phase	25
3.5	Morphing spectrale	27
3.6	Morphing visuel	30
4	Implémentations artistiques	32
4.1	<i>Introduction</i>	32
4.2	Le vocodeur de phase - <i>Une capacité sans fin</i>	32
5	Conclusion	35
5.1	Résumé de la recherche	35
5.2	Applications	35
5.3	Recherche pour l'avenir	36

Table des figures

2.1	Complex DFT	8
2.2	Circle de la transformation Fourier	9
2.3	Magnitude et phase	10
2.4	Overlapping	12
2.5	Interpolation du facteur α	19
3.1	Pitch Tracking	21
3.2	Calcul de la magnitude Dominante dans un FFT	24
3.3	Multiple Pitch tracking	24
3.4	Calcule des Fréquences	25
3.5	Le vocodeur de phase	28
3.6	Fenetrage gaussien	29
3.7	Morphing en temps réel	29
3.8	Visual Morphing	30
4.1	Super Phase Vocoder I	33
4.2	Super Phase Vocoder II	34

Chapitre 1

Introduction

1.1 À propos

Cette recherche aura pour objectif de décrire tous les éléments du processus du morphing sonore en temps réel, sous un angle à la fois technique et artistique. L'exploitation des notion spectrales concernant le morphing, permettra une connaissance profonde sur ce domaine. Ce propos déverrouillera des nouvelles méthodes pour manipuler un vocodeur de phase ou tout qui peut fonctionner comme un mécanisme central du morphing en temps réel dans cette recherche.

Àfin d'analyser le processus du morphing sonore incluant en parallèle des données mathématiques, informatiques, artistiques et musicales nous garderons une perspective globale et abrégé sur notre matière, qui est le morphing.

La partie informatique est presque entièrement réalisée sur le logiciel MaxMSP et Jitter. L'utilisation des ressources de MaxMSP pour l'analyse spectrale et la transformation Fourrier, ainsi que l'assistance de Jitter pour les processus visuels, nous permettront aussi de découvrir les vastes capacités du logiciel.

Le texte est réalisé en LaTeX pour des raison pratiques concernant la syntaxe clair, des formules mathématiques, un format plus efficace, et la partage de ce mémoire.

Les archives de cette recherche sont disponibles pour le public sur "Github.com" ou les chercheurs, les musiciens et tous ceux qui sont intéressés ont le pouvoir d'accéder les données, le code, les fichiers sonore et les exemples artistiques. Le lien pour le repository Github est le suivant <https://github.com/melkisedeath/Un-Guide-sur-l-analyse-Spectrale>

1.2 Structure

Cette recherche se compose de trois parties : la première, aborde le contexte mathématique et théorique de l'analyse spectrale et, par extension, la transformation de Fourier ; la deuxième partie, concerne la progression numérique, en réalisant l'analyse spectrale et le morphing sur le logiciel MaxMSP ; la troisième partie, propose des applications artistiques, en utilisant des effets spectraux, c'est-à-dire du vocodeur de phase jusqu'au morphing.

Dans la première partie, on traite de la transformation spectrale d'un signal. Il est essentiel pour cette partie de se baser sur la transformation de Fourier. Notre approche utilise l'algorithme FFT¹ de la TFD² , comme c'est un algorithme qui s'applique facilement au domaine digital . En revanche, les autres moyens pour calculer la transformation de Fourier seront aussi analysés. La partie technique inclut aussi des termes comme fenêtrage et filtrage Gabor. Par la suite, on se focalisera sur l'analyse spectrale, on décomposera les composants du spectre et on recherchera comment le paramétrage de nombreuses variables changeront notre résultat. À partir de ce moment, on pénétrera dans le domaine du vocodeur de phase.

La deuxième partie, concerne l'implémentation de la théorie de la première partie sur le logiciel MaxMSP. MaxMSP est un logiciel fait pour la programmation musicale et, depuis 2002, la partie du logiciel « Jitter » permet aux utilisateurs de manipuler des informations multidimensionnelles. MaxMSP utilise FFT de FTD pour réaliser la transformation de Fourier.

Dans cette partie, on construira un vocodeur de phase dans Max qui permettra de visualiser et de manipuler les composants du spectre. De plus, on va créer des patches sur Max qui nous

1. La transformation de Fourier rapide
2. La transformation de Fourier discrète

permettront de faire du morphing sonore et visuel, de même que du morphing de la forme. Très précisément, le morphing visuel s'applique aux images (données bidimensionnelles) et le morphing de la forme se réfère aux données 3D.

Dans la dernière partie de cette recherche, on va créer une série d'implémentations artistiques des patches techniques qu'on a construit dans la deuxième partie.

1.3 MaxMSP et Jitter

MaxMSP est un logiciel, créé à l'origine par Michael Puckette et acheté par la suite par la société américaine Cycling74. Le cœur du logiciel utilise les langages C++ et Javascript, mais le code de base s'exécute dans un environnement plutôt graphique pour la facilitation de l'utilisateur.

Le langage MaxMSP fonctionne avec des objets qui rappellent essentiellement un type de fonction simple. Les objets peuvent être manipulés graphiquement et l'utilisateur peut connecter plusieurs fonctions ou objets, afin de créer un patch complexe. C'est un langage en temps réel qui a des possibilités illimitées. Il est structuré à partir de trois parties, Max contient essentiellement les processus logiques, alors que MSP permet le traitement du signal et Jitter celui de l'édition d'image et de vidéo. Max6 and Max7 inclut également la conception 3D dans OpenGL et principalement GLSL.

Dans le domaine de l'analyse et des traitements spectraux, un processus apparemment complexe de conversion des données de volume en deux axes représentant la localisation du vecteur d'intensité au niveau cartésien est utilisé. Avec cette conversion, il est possible d'analyser plus de variables telles que la fréquence, l'intensité et la durée des fréquences. En substance, il s'agit d'une forme de transformation de Fourier rapide (FFT).

Le logiciel est déjà connu, des grands entreprises et des institutions qui focalisent leur recherche sur MaxMSP. Cycling74, l'Ircam et d'autres universités ont créé des algorithmes qui rendent le traitement d'un spectre possible et de faciliter l'utilisation même par le plus amateur. Certaines bibliothèques qui offrent ces fonctionnalités sont SuperVP, Max SoundBox et des outils

similaires.

1.4 Analyse spectrale

L'analyse spectrale est basée à l'origine sur la théorie de Fourier, écrite par Jean-Baptiste Joseph Fourier. Elle concerne les propriétés thermodynamiques des matériaux. Dans cette théorie, il est connu que chaque spectre complexe peut être interprété comme un mélange de fréquences uniques³. Chaque son complexe harmonique peut être ré-énoncé à partir d'une somme de sinusoïdes simples d'amplitudes différentes. Le processus inverse sera utilisé, plus tard dans un domaine musical, par des compositeurs, comme une technique musicale qui a été décrite comme la synthèse additive.

La fonctionnalité de l'analyse spectrale et donc de la transformation Fourier présente des vastes potentiels, et parmi eux des applications musicales. Dans le cadre de cette recherche on focalisera sur le vocodeur de phase, un outil qui à partir de la transformation Fourier permet une variété des effets sonores et musicales. Cet outil permettra une manipulation sonore pour changer la hauteur, changer le rythme de la lecture, affecter le timbre sonore ou encore interpoler (morpher) entre deux sons.

1.5 Morphing

La notion de morphing est très populaire sur des données de l'image, des dessins et de la musique également. Pour autant, il existe deux grandes catégories, le morphing sonore et le morphing visuel.

3. Jean Joseph Baptiste Fourier. *De la Chaleur*. Paris, 1822.

1.5.1 Morphing sonore

En avançant sur le terrain de l'analyse spectrale pour avoir la manipulation sonore et la création des effets musicaux, le vocodeur de Phase est l'outil que nous utiliserions. Le vocodeur de phase nous permet de réaliser des processus spectraux en temps réel sans perte de qualité sonore et en utilisant un minimum des données de calcul.

A l'origine, le vocodeur de phase est un processus spectral qui utilise les données de phase de chaque index⁴ d'une fenêtre pour calculer sa fréquence. Les définitions de ces notions seront présentées en détail dans les prochaines parties de la recherche. J'ai choisi de l'utiliser dans cette approche car les outils actuels du morphing en temps-réel développés par Ircam(SuperVP) sont construits sur la structure de base du Vocodeur de Phase.

Dans notre propre vocodeur de phase nous réaliserons l'effet du morphing spectrale en temps réel.

1.5.2 Morphing visuel

La compréhension du morphing visuel est relativement plus simple pour deux raisons : i) les images sont des données statiques par rapport au son qui est un phénomène dynamique, c'est à dire que le son est lié au temps. ii) Le résultat d'un morphing visuel est interprété par la vision, tandis que le son dépend de l'audition.

4. index : l'image du son dans un instant précis

Chapitre 2

Context théoristique

2.1 Introduction

Dans cette section de notre recherche, nous allons développer toutes les notions techniques d'un point de vue mathématique(théorique) et musicale

Une méthode classique de synthèse d'un son complexe variant dans le temps, consiste à combiner plusieurs formes d'onde élémentaires. Les formes d'onde superposées dans la synthèse additive sont souvent sinusoïdales. Dans certaines conditions, les sinusoïdes individuels fusionnent et le résultat est perçu comme un seul son riche.

L'idée derrière cette méthode n'est pas nouvelle. En effet, la synthèse additive est utilisée depuis des siècles dans des instruments traditionnels tels que l'organe.

Lorsqu'un son quasi-périodique est analysé, son énergie spectrale est concentrée autour de quelques fréquences discrètes (harmoniques). Ces lignes de fréquence correspondent à différents signaux sunisoïdaux appelés partiels. L'amplitude de chaque partiel n'est pas constante et sa variation temporelle est critique pour la caractérisation du timbre. Et plus particulièrement, dans la phase initiale de transistors (attaque) d'une note. La fréquence de chaque composant peut cependant être considérée comme variant lentement.

La synthèse additive est la somme des oscillateurs sinusoïdaux dont l'amplitude et la fréquence varient dans le temps. Les paramètres de contrôle sont déterminés par analyse spectrale.¹

2.2 Le décodage mathématique

2.2.1 La transformation Fourier

La transformation de Fourier, publié en 1822 par Jean Baptiste Joseph Fourier, est une vaste théorie qui au départ servait à décrire les capacités des matériaux métalliques. Cette théorie a essentiellement prouvé, que la décomposition de quelques fonctions peut être représentée comme une somme infinie de sinusoïdes.

Dans le terrain acoustique et puis musical les composantes de la transformation de Fourier forment le spectre d'un son. L'analyse spectrale est essentiellement le passage du son du domaine temporel au domaine fréquentiel. Au vu de ces propos, l'analyse spectrale, est une transformation Fourier suivie de quelques processus mathématiques, permettant de visualiser les fréquences qui composent un son

Il existe plusieurs façons de calculer la transformation de Fourier discrète (TFD), dont la transformation de Fourier rapide (FFT). Bien qu'il donne le même résultat que les autres approches des conversions de Fourier, elle est néanmoins la plus efficace par la puissance de calcul qu'il requiert. Il est à noter que la FFT est l'un des algorithmes les plus courants dans le traitement du signal car son exécution ne nécessite généralement que quelques lignes de code. Cependant, c'est l'un des algorithmes les plus complexes de DSP².

FFT est un algorithme complexe, dont la compréhension approfondie est faite par les mathématiciens. La FFT est basée sur l'algorithme, qui est appelée Complex TFD. La compréhension de l'algorithme nécessite l'utilisation de nombres complexes. La TFD complexe se réfère précisément à cela, en utilisant des nombres complexes.

1. Curtis Roads. *The computer Music Tutorial*, 1996.

2. Digital Signal Processing

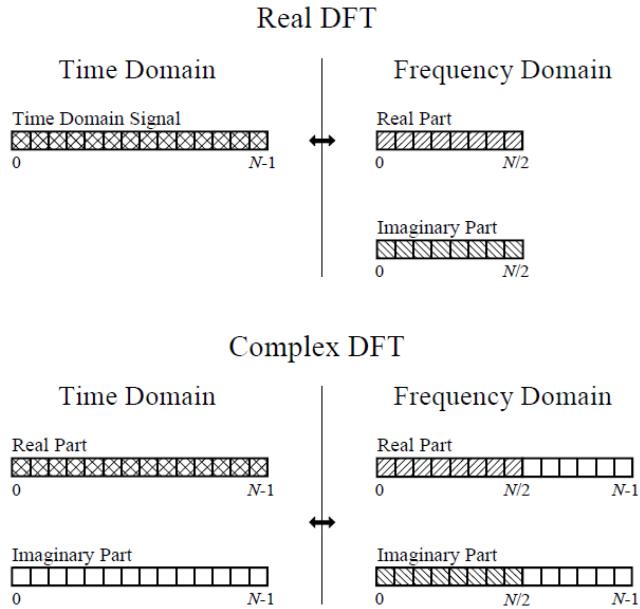


FIGURE 2.1 – Complex DFT

La figure 2.1 justifie graphiquement ce processus. Dans la colonne de gauche, on peut voir la représentation du signal dans le domaine temporel. Tandis que, dans la colonne de droite est placée sa convention FFT, dans le domaine fréquentiel. N est la taille de la fenêtre des données DSP, ou la durée totale en termes généraux. Comme le montre la figure 2.1, le signal est décomposé en deux variables d'axe, une réelle et une imaginaire.

En mathématiques avancées, les domaines temporels et fréquentiels contiennent chacun un signal constitué de N points, où N est un nombre complexe. Chacun de ces points est composé de deux parties, la partie réelle et la partie imaginaire. Par exemple, lorsque nous nous référons à l'échantillon complexe $x[42]$, il s'agit de la combinaison $\text{Re}(x)[42]$ et $\text{Im}(x)[42]$. En d'autres termes, chaque variable a deux nombres. Dans cette tentative, lorsque deux variables complexes sont multipliées, les quatre composants doivent être combinés pour former les deux composants du produit. Pour comprendre la FFT, cette notation de nombres complexes est utilisée.

FFT fonctionne en décomposant un intervalle de temps N dans les variables de signal N -temps consistant en un point unique sur le champ cartésien. La deuxième étape consiste à calculer les N fréquences de données correspondant à ces N points de temps. Enfin, les N -points de la fréquence résultante le spectre sont synthétisés dans un seul spectre³.

3. Steven W. Smith. The scientist and engineer's guide to digital signal processing. California Technical

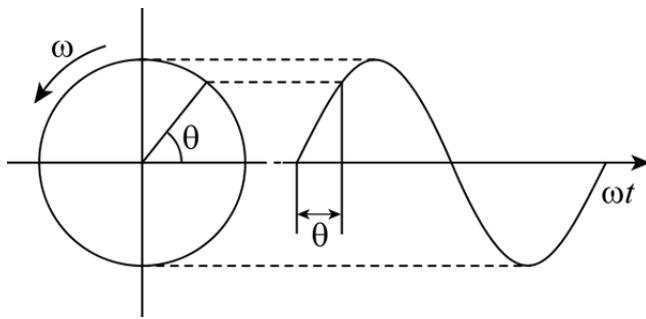


FIGURE 2.2 – Circle de la transformation Fourier

La fonction périodique de Fourier peut être formée comme ci-dessous :

$$^4x(t) = c_0 + \sum_{n=1}^{\infty} c_n \cos(\omega t + \theta_n) \quad (2.1)$$

Où t est le temps, x le signal sonore, c_0 l'harmonique fondamentale, f la fréquence, $\omega = 2\pi f$ et θ_n la phase de chaque harmonique. Dans cette formulation de la transformation Fourier il est clair que un son x dépendant de temps t peut être interprété comme une somme des sinusoïdes. En revanche, la formule générale de la transformation Fourrier est la suivante :

$$^5F[x(t)](\omega) = \int_{-\infty}^{+\infty} e^{-j\omega t} x(t) dt \quad (2.2)$$

Où $F[x(t)](\omega)$ est la transformation Fourrier du signal x dépendante à une fréquence variée ω . Dans le domaine sonore cette fréquence se varie entre 0 et le SR⁶.

Une syntaxe mathématique de type $e^{j\pi t}$ où j est un nombre complexe rends graphiquement les données au fonction d'un cercle dans le plan cartésien (réel - complexe). Sur la transformation Fourier on rend le signal $x(t)$ autour d'un cercle ($e^{-j\omega t}$), avec une fréquence f , puisque $\omega = 2\pi f$, dans le sens des aiguilles, donc $-j$. Le terme $e^{j2\pi f}$ s'appelle l'exponentiel complexe et peut être

Publishing, 1997

4. Curtis Roads. The Computer Music Tutorial. MIT Press, Cambridge, MA, USA, 1996. ISBN 0262680823.[p. 1085]

5. Hermann L F. Helmholtz. The Sensations of Tone. New York, 1895.[p. 215]

6. SR = Sampling Rate ou fréquence d'échantillonage en français. Suivant 44100 ou 48000Hz

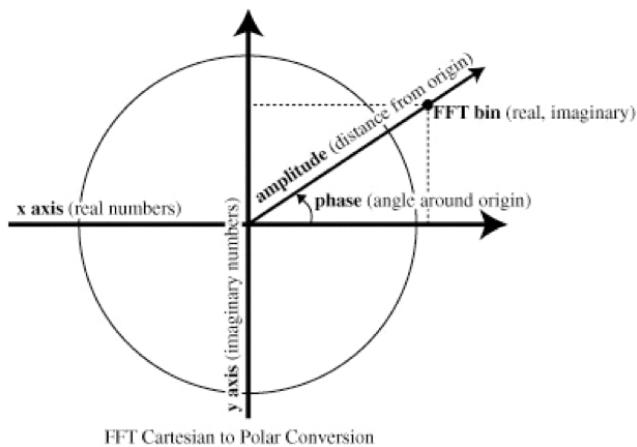


FIGURE 2.3 – Magnitude et phase

décrit comme :

$$7e^{j2\pi f} = \cos(2\pi f) + j \sin(2\pi f) \quad (2.3)$$

Les données produites de l'exponentiel complexe sont cartesiennes. Afin d'améliorer la manipulation de ces données il suffit de les changer en forme polaire.

Après la transformation Fourier de notre signal, il y a deux facteurs polaires à manipuler, la phase et la magnitude. La phase de notre signal conclut les informations de fréquence, ou la magnitude les informations d'amplitude. La phase et la magnitude sont calculées pour chaque échantillon de notre fenêtre de la FFT. Elles sont dérivées par les coordonnées réelles et imaginaires. Ou la magnitude égale : $m(x) = \sqrt{i(x)^2 + r(x)^2}$ et la phase $\theta(x)$ est l'angle de la représentation graphique de la magnitude sur le plan cartésien de coordonnées cartésien ou $\theta(x) = \tan^{-1}\left(\frac{i(x)}{r(x)}\right)$. 2.3

Dans cette recherche, il est important d'utiliser une formule qui soit dépendante du temps

7. C. Roads, S.T. Pope, A. Piccialli, and G. De Poli. Musical Signal Processing. Taylor & Francis, 2013.
ISBN 9781134379705.

continue, donc d'un fenêtrage. Cet algorithme est la STFT⁸ :

$$^9X(\omega, \tau) = \sum_t^{\infty} x(t)\omega(t - \tau)e^{-j\omega t} \quad (2.4)$$

Où x est le signal, X sa transformée Fourier (une abréviation de la forme $F[x(t)]$), $\omega = 2\pi f$, t le temps continu, τ l'instant temporel, c_n l'harmoniques, $\omega(t)$ le fenêtrage, et j un nombre complexe.

Dans cette recherche, on va utiliser la forme continue TFD et puis FFT, vu que cette formule est premièrement utilisée dans MaxMSP et en plus requiert une puissance calculatrice plus efficiente que d'autres méthodes :

$$^{10}X(\omega) = \frac{1}{N} \sum_{n=0}^{N-1} x(n+1)e^{-j\frac{\omega n}{N}} \quad (2.5)$$

Où N est la totalité des instants sonores échantillonnés de notre signal x , n signifie chauque instant sonore, et $\omega = 2\pi f$ est toujour la frequence variée.

Pour chaque transition au domaine frequentiel, il faut une tranformation retour au domaine temporel. Le resultat est caracterisé comme la tranformation Fourier inverse (IFFT).

$$^{11}x(n) = \frac{1}{N} \sum_{f=0}^{N-1} X(f)e^{j\frac{2\pi fn}{N}} \quad (2.6)$$

2.2.2 Fenêtrage et filtres Gabor

Comme il a été déjà mentionné, pour calculer une STFT, une fenêtre d'une certaine durée est nécessaire pour être définie. Pour cette fenêtre, une certaine fonction peut être appliquée pour

8. STFT : Short Time Fourier Transform,

Jown Strawn. Introduction to digital sound synthesis. Digital Audio Engineering, pages 141– 134, 1985.

9. Tadej Droljc. STFT Analysis Driven Sonographic Sound Processing in Real-Time using Max/MSP and Jitter. 2011.

10. Jean-François Charles. A tutorial on spectral sound processing using maxmsp and jitter. Computer Music Journal, pages 87–102, Printemps 2008.

11. Alan V. Oppenheim et Ronald W. Schafer. Discrete-time signal processing. Prentice Hall Press.

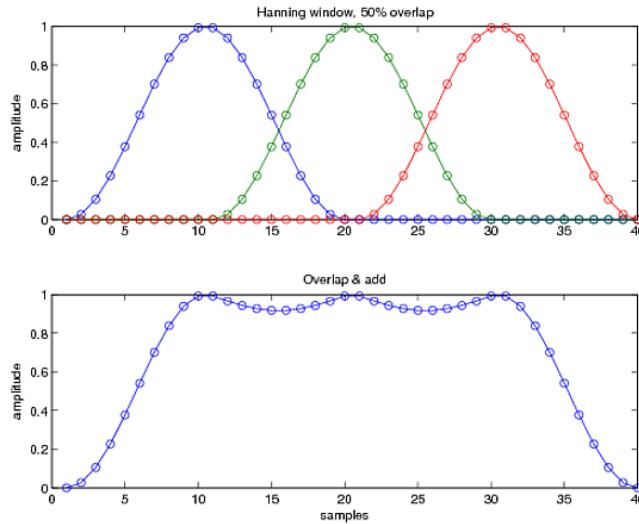


FIGURE 2.4 – Overlapping

éviter les cliques comme (hanning, hamming, exponentielle, etc). Pendant STFT, il est habituel de chevaucher les fenêtres l'un dans l'autre. Cette procédure est expliquée graphiquement dans la figure 2.4.

Cette technique, nommée overlapping, nous permet de créer un résultat lisse au niveau sonore. Lors de changement de fenêtres l'auteur ne peut pas percevoir que le son était décomposer en morceaux mais il attend un résultat sonore unifié¹².

Par la suite, nous allons examiner la possibilité d'appliquer un fenêtrage Gaussien à notre transformation Fourier en utilisant la transformation Gabor :

$$G_x(t, f) = \int_{-\infty}^{+\infty} e^{-\pi(\tau-t)^2} e^{-j\omega\tau} x(\tau) d\tau \quad (2.7)$$

Pour manipuler la courbe du filtre Gabor, il faudrait changer les paramètres. La formule normalisée devient alors :

$$G_x(t, f) = \int_{-\infty}^{+\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{\tau-t}{\sigma})^2} e^{-j\omega\tau} x(\tau) d\tau \quad (2.8)$$

12. Daniel W. Griffin et Jae S. Lim. Signal estimation from modified short-time fourier transform. IEE Transaction on acoustics, speech, and signal processing, ISSE Vol. 32 :236–243, Avril 1984.

Où σ est le paramètre de la courbe gaussienne. Un filtrage Gabor nous permettra de créer un fenêtrage plus équilibré qui lisse le résultat sonore après l'analyse .

La même logique peut être validée pour n'importe quel genre de fenêtrage (hanning, hamming, etc.).

2.2.3 Le vocodeur de phase

Définition

Le vocodeur de phase est un système d'analyse-synthèse qui a pour date intermédiaire le spectre de Fourier discret, variant dans le temps du signal d'entrée. Il peut être formulé de telle sorte que le signal synthétisé soit identique à l'original, à la fois théoriquement et pratiquement. Les données intermédiaires peuvent être transformées, même sans perte d'information, en une représentation d'amplitude et de fréquence plus conventionnelle. Ces données intermédiaires peuvent ensuite être utilisées pour resynthétiser la tonalité à des différentes hauteurs ou à des vitesses différentes de l'original. Le vocodeur de phase n'a pas de telles restrictions et peut tout aussi bien gérer le vibrato et les inharmonicités des sons¹³.

Histoire

Le vocodeur de phase a été introduit, en 1966, par Flanagan comme un algorithme qui préserverait la cohérence horizontale entre les phases des segments qui représentent les composantes sinusoïdales. Ce vocodeur de phase original ne tenait pas compte de la cohérence verticale entre les intervalles de fréquence adjacents, et par conséquent, l'étirement temporel de ce système produisait des signaux sonores qui manquaient de clarté. La reconstruction optimale du signal sonore de STFT après modifications d'amplitude a été proposée par Griffin et Lim¹⁴. Cet algorithme ne considère pas le problème de produire un STFT cohérent, mais il permet de trouver

13. Johannes Grünwald. Theory, implementation and evaluation of the digital phase vocoder in the context of audio effects, 2010.

14. Daniel W. Griffin et Jae S. Lim. Signal estimation from modified short-time fourier transform. IEE Transaction on acoustics, speech, and signal processing, ISSE Vol. 32 :236–243, Avril 1984.

le signal sonore dont le STFT est aussi proche que possible d'une STFT modifiée, même si la STFT modifiée n'est pas cohérente (ne représente aucun signal).

Le problème de la cohérence verticale est demeuré un enjeu majeur pour la qualité des opérations de mise à l'échelle temporelle jusqu'en 1999, lorsque Laroche et Dolson¹⁵ ont proposé un moyen de préserver la cohérence des phases dans les compartiments spectraux. La proposition de Laroche et Dolson doit être considérée comme un tournant dans l'histoire des vocodeurs de phase. Il a été montré que, grâce à la garantie d'une cohérence de phase verticale, des transformations d'échelle temporelle de très haute qualité peuvent être obtenues.

L'algorithme proposé par Laroche n'a pas permis de conserver la cohérence de la phase verticale pour les amorces sonores (début des notes). Une solution à ce problème a été proposée par Roebel¹⁶. Un exemple de mise en œuvre, dans un logiciel, de la transformation de signal basée sur un vocodeur de phase utilisant des moyens similaires à ceux décrits ci-dessus, pour obtenir une transformation de signal de haute qualité, est le SuperVP de l'Ircam.

Une approche à la description du vocodeur de phase consiste à représenter le signal via la succession des images successives de une transformée de Fourier Discrète (TFD) de une fenêtre de longueur N. Ces images sont d'abord multipliées par une fenêtrage appropriée (telle que Hamming, Hanning, Kaiser, Blackman, etc.) puis transformé en Fourier dans le domaine fréquentiel. A ce stade, toute modification prudente du spectre peut être faite, avant la transformation inverse au domaine temporel avec la transformée de Fourier discrète inverse (TDFI). Là, les parties d'overlap et éventuellement fenêtrées sont additionnées, ce qui donne le résultat final.

La STFT (une transformation caractérisée des successions des TFD) d'un signal fenêtré est défini comme suit :

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X_f(n) W_N^{nk}, \quad \forall n \in SR \quad (2.9)$$

15. Mark Dolson. The phase vocoder : a tutorial. Computer Music Journal, pages 14–27, Printemps 1986.

16. Axel Roebel. Morphing sound attractors. In 3rd. World Multiconference on Systemics, Cybernetics and Informatics (SCI'99) and the 5th. Int'l Conference on Information Systems Analysis and Synthesis (ISAS'99), 1999, Florida, United States, Proc. of the 3rd. World Multiconference on Systemics, Cybernetics and Informatics (SCI'99), 5th. Int'l Conference on Information Systems Analysis and Synthesis (ISAS'99), 1999.

ou $W_N = e^{\frac{2\pi j}{N}}$ et $h(n)$ est une fenêtre approximativement choisie.

$$h_f(n) = \frac{1}{N} h(n) W_N^{nf}, \quad k = 0, 1, \dots, N-1 \quad (2.10)$$

Pendant le processus de l'analyse, la succesion des images d'une STFT font parties du signal d'entrée $x(n)$, sur la position $n_a^u = uR_a$, où R_a est nommé *input hop size* et u doit être un entier. La fenêtre (ou la durée) de la DFT est définie par la taille N , où le *hop size* est forcément une sous-multiplication de N . Cet effet permettra de réaliser un *overlap* de 50%, 75%, etc.

$$X[n_a^u, \omega] = \sum_{n=0}^{N-1} \tilde{x}_u(n) e^{-j\omega \frac{n}{N}} \quad (2.11)$$

$$\text{ou} \quad x_u(n) = h_a(n)x(n - n_a^u)$$

Le terme $\tilde{x}_u(n)$ propose l'estimation des fenêtres symétriques calculées. Donc si nous supposons un overlap de fenétrage par 50 %, ou bien $N/2$, puis une manière d'éviter les assymétries de phase est composée ci-dessus :

$$\tilde{x}(n) = x[((n - N/2))_N] \quad (2.12)$$

ou $((.))_N$ présente l'opération du modulo et N est la durée de séquence et il devait un pair.

Donc la transformation de Fourier du signal liée au fenétrage devient :

$$X(rl, f) = \sum_{N=0}^{N-1} x(n)h(rl - n)e^{-j\frac{2\pi}{N}fn} \quad (2.13)$$

C'est, donc, une fonction de deux variables discrètes, le temps rl et la fréquence f . L'index rl est la position de la fenêtre, r étant le numéro d'image (frame) et l la taille du décalage de la fenêtre d'analyse.

$X(rl, f)$ peut être vu comme le spectre de la multiplication $x(n)h(rl - n)$, qui est le signal d'entrée $x(n)$ multiplié par la fenêtre décalée à la position rl . Ce n'est pas le spectre exact, mais sa convolution avec la transformation de Fourier de la fenêtre.

La procédure standard de *overlap* consiste à additionner les buffers et à les diviser par la somme des fenêtres décalés. Le signal de sortie $y(n)$ est exprimé comme :

$$y(n) = \frac{\sum_r \bar{x}(rl, n)}{\sum_r h(rl - n)} \quad (2.14)$$

Pour construire la reproduction du signal d'origine ou du signal transmuté après traitement, un iFFT est nécessaire. Dans le vocodeur de phase, cette procédure accède aux informations de phase et d'amplitude et se forme comme suit :

$$\bar{s}(rl, k) = \frac{1}{N} \sum_{k=0}^{N-1} |\bar{S}(rl, k)| e^{j(\frac{2\pi}{N} km + \theta(rl, k))} \quad (2.15)$$

Enfin pour calculer la fréquence de chaque image instantanée il suffit de suivre la formule :

$$\bar{f}_{k,r} = \frac{\theta(rl, k) - \theta((r-l)l, k)}{l} \quad (2.16)$$

2.2.4 Applications du vocodeur de phase

Time Stretching

Pour réaliser un étirement du temps d'un son, traditionnellement, il suffit de baisser ou augmenter le rythme de sa lecture, cela produit par ailleurs le changement de la hauteur. Le vocodeur de phase permet d'effectuer un étirement temporel sans pour autant changer la hauteur ou la qualité sonore. Afin de mieux comprendre le fonctionnement du vocodeur il faut considérer la transformation Fourier à court terme, pour un étirement temporel. Pour visualiser le résultat on peut imaginer, que pour chaque période de temps de la transformation Fourier appliquée,

une série des harmoniques sauvegardées dans une fenêtre, et le facteur de son overlap. Il suffit d'éloigner ou rapprocher les fenêtres, pour changer le rythme de la lecture sans affecter la hauteur sonore.

Transposition de la hauteur

Comme le vocodeur de phase peut être utilisé pour réaliser un étirement temporel d'un son sans affecter sa hauteur, il devrait également être possible de faire l'inverse, c'est-à-dire changer la hauteur sans changer le rythme de la lecture. En effet, cette opération est facilement accomplie. La procédure est de changer le rythme de la lecture par le facteur de changement de la hauteur souhaitée, puis de jouer le résultat sonore produit à la fréquence d'échantillonnage «incorrecte». Par exemple, pour augmenter la hauteur d'une octave, le son est d'abord agrandi d'un facteur de deux, et il est ensuite joué à deux fois l'original taux d'échantillonnage.¹⁷

Freeze

À cet effet, nous prenons un certain fenêtre d'analyse à partir du son sélectionné et nous «gèlons» ce son dans le temps. Pour achever cet effet il s'agit de rendre le rythme de la lecture de notre vocodeur de phase au zero. Cela peut se comparer à un étirement sonore infini. On peut ainsi appliquer les mêmes principes d'un étirement sonore normal.

Robotisation

Pour faire une robotisation d'un signal il faut mettre la phase de chaque échantillon à zéro. Cet effet résulte à un son robotisé et métallique.

17. Mark Dolson. The phase vocoder : a tutorial. Computer Music Journal, pages 14–27, Printemps 1986.

Harmonisation - chuchotement

Pour réaliser cet effet on doit donner une valeur aléatoire soit à la phase, soit à la magnitude de chaque échantillon de la fenêtre de la FFT¹⁸.

Morphing

La définition générale du morphing consiste à combiner deux (ou plusieurs) éléments distincts en une seule entité qui contient les deux éléments. Le processus de morphing dépend généralement d'une seule variable, appelée un facteur de morphing ou une interpolation. Ce processus dépend, par ailleurs, du facteur temporel puisque le morphing est un phénomène dynamique.

$$M(\alpha, t) = \alpha(t)\widehat{S}_1 + [1 - \alpha(t)]\widehat{S}_2 \quad (2.17)$$

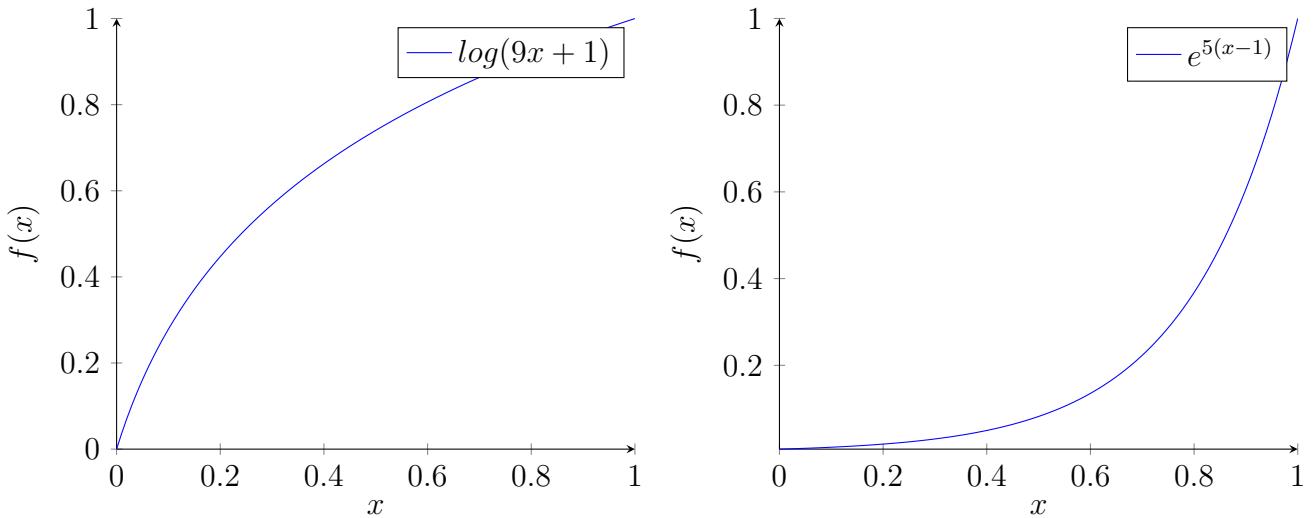
Ou $\alpha(t) \in [0 : 1]$

La manipulation du paramètre α nous permettra de changer la dynamique du morphing. La question qui se pose c'est pourquoi se contenter d'une manipulation linéaire ? Nous proposons alors d'effectuer une manipulation de α sur une courbe exponentielle ou logarithmique(ex. sur la figure 2.5).

La différence fondamentale entre le morphing d'une image, est le rapport a la variable temporelle. Le son est un phénomène dynamique, donc il ne peut pas être interpolé linéairement. Le terrain du morphing sonore nécessite des fonctions multiples pour atteindre un résultat satisfaisant. Pour obtenir un morphing sonore il faut calculer l'enveloppe spectrale de notre FFT.

18. Johannes Grünwald. Theory, implementation and evaluation of the digital phase vocoder in the context of audio effects, 2010.

19. Axel Roebel. Morphing sound attractors. In 3rd. World Multiconference on Systemics, Cybernetics and Informatics (SCI'99) and the 5th. Int'l Conference on Information Systems Analysis and Synthesis (ISAS'99), 1999, Florida, United States, Proc. of the 3rd. World Multiconference on Systemics, Cybernetics and Informatics (SCI'99), 5th. Int'l Conference on Information Systems Analysis and Synthesis (ISAS'99), 1999.

FIGURE 2.5 – Interpolation du facteur α 

Pour avoir un résultat proche du son d'origine il s'agit de calculer le Cepstre. Le cepstre résulte en analogie de la transformation Fourier inversée :

$${}^{20}C(t) = \sum_{n=0}^N \log(|X(n)|) e^{j \frac{2n\pi t}{N}} \quad (2.18)$$

Il existe des approches différentes du Morphing spectrale, mais dans cette recherche nous nous focaliserons sur le morphing en temps réel, et plus spécifiquement sur le morphing avec un vocodeur de phase.

20. Fernando Villavicencio et Xavier Rodet Axel Roebel. On cespstral and all-pole based spectral envelope modeling witg unknown model order. Pattern Recognition Letters, (28) :1343–1350, 2007.

Chapitre 3

Design

Dans ce chapitre nous allons réaliser les théories discutées, dans le chapitre précédent (No. 2). La réalisation des notions théoriques vont être implémentées dans le logiciel MaxMSP. L'implémentation de la théorie nous permettra de la comprendre profondément et aussi de la découvrir

3.1 MaxMSP

Définition de MaxMSP et introduction sur le logiciel

————commentaire————

3.2 Pitch tracking

L'analyse spectrale est essentiellement un outil qui permet de relever les composants d'un son et, par conséquent, les fréquences et les amplitudes de ses harmoniques¹. En ce qui concerne la détection de la fréquence dominante, j'ai implémenté le patch présenté sur la figure 3.1.

1. Dolson, 1986

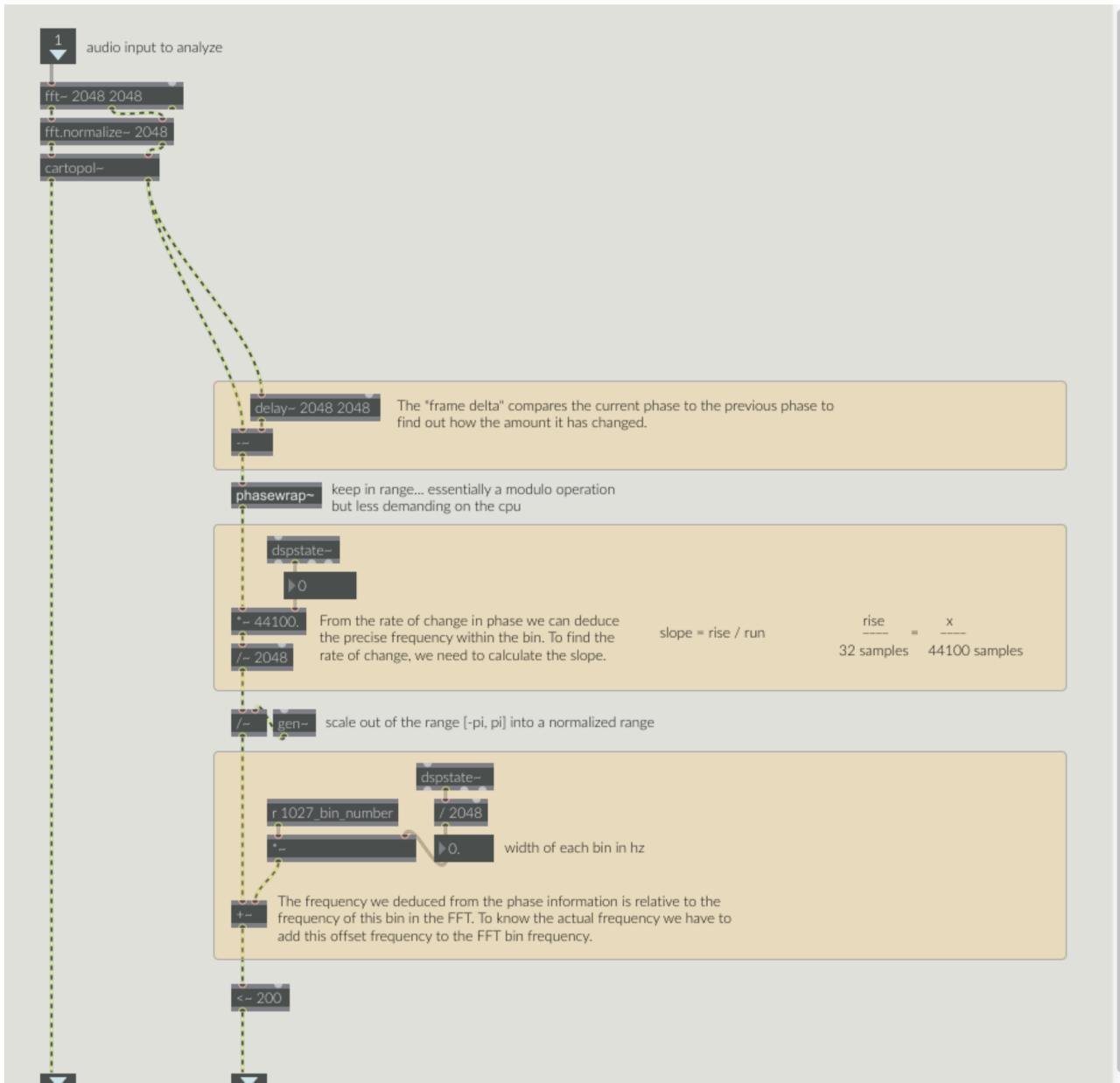


FIGURE 3.1 – Pitch Tracking

Le premier objet utilisé, appelé `fftin` , transforme le signal en domaine fréquentiel. Il s'agit essentiellement d'une fonction qui effectue une transformation de Fourier rapide (FFT) et donne trois sorties. Il est nécessaire de déclarer à chaque fois qu'une fenêtre représentant le nombre d'échantillons négligés dans une analyse de Fourier. Le deuxième aspect à déclarer, est l'enveloppe de la fenêtre afin d'éviter les cliques et les pics. Les deux premières sorties représentent le vecteur d'amplitude dans un système d'axes cartésien bidimensionnel. La troisième sortie permet de fournir l'index de la corbeille d'échantillons. Pour poursuivre la procédure, il est crucial de normaliser les données, afin que le flux de données réel ne soit pas hors de la plage souhaitée. Ce processus est réalisé avec l'aide de l'objet `fft.normalize`. Il est cependant toujours nécessaire de déclarer la taille de la fenêtre comme avant.

Les données sont traitées après leur transformation en informations plus utilisables. Ainsi, avec l'objet `cartopol` , le vecteur de l'amplitude est transformé en sa forme polaire. Cette transformation se produit pour chaque index de bin. Cet objet sort la magnitude et la phase de chaque index.

Ensuite, une fonction appropriée est nécessaire pour déterminer quelle corbeille contient les composantes de fréquence les plus fortes. Pour construire cette fonction, deux entrées sont nécessaires, la phase et son numéro d'index. L'utilisation d'un objet `gen` est préférée pour la construction d'une fonction personnalisée à l'intérieur de Max. Bien sûr, avant de faire cela, la taille de la fenêtre doit être déclarée.

Il y a un objet dans Max qui permet d'utiliser comme entrée de code, appelée `codebox`. Tout comme le code C, il est obligatoire de déclarer les variables avant de passer au codage. Les variables dans ce cas sont : magnitude, index et last, ainsi que le taille de la fenêtre déclaré au début. Ce processus nécessite deux boucles IF imbriquées. Pour chaque bin du son, ce processus sera répété et il déterminera la fréquence dominante de chaque fenêtre. Dans la boucle IF imbriquée, il est déterminé si chaque index est plus fort que le précédent et le résultat est retourné. La dernière étape consiste à déterminer si le nombre d'index atteint la limite de taille de trame, afin de terminer le processus et de réinitialiser les variables.

Pour sortir l'amplitude du partiel le plus fort, il est nécessaire de créer un sample holder qui

n'émet l'amplitude que lorsque le contrat de gen est vrai. L'objet sah (sample and hold) sort les valeurs de la trame FFT qui est la trame avec la plus grande amplitude. Afin de déterminer l'amplitude et la phase désirée, deux objets sont nécessaires. Ensuite, une fonction «frame delta» compare la phase en cours à la phase précédente, pour que l'utilisateur comprenne comment le montant a été changé.

L'étape suivante pour déterminer avec succès la fréquence la plus forte du bin est d'utiliser un objet phasewrap qui fonctionne essentiellement comme une opération modulo, mais moins exigeant sur le processeur. A partir du taux de changement de phase, on peut en déduire la fréquence précise à l'intérieur du Bin. Pour trouver le taux de changement, nous devons calculer le Slope. Le slope est égale $rise * run$, où rise est la différence de la phase entre le courant et le dernier échantillon, et la Run est la taille du bin. Essentiellement, la fréquence est égale à l'augmentation, multipliée par la fréquence d'échantillonnage, divisée par la taille ou l'exécution. L'expression entière est ensuite divisée par deux pi pour étendre dans une gamme plus normalisée. La fréquence déduite de l'information de phase est relative à la fréquence de cette case dans la FFT. Pour obtenir la fréquence réelle, la fréquence de décalage doit être ajoutée à la fréquence du FFT bin.

La dernière étape consiste à convertir l'amplitude analogique en numérique et à faire deux rampes pour la fréquence et l'amplitude. Cette procédure lisse et stabilise les résultats.

Dans la figure 3.2, on retrouve l'index de chaque fenêtre qui possède l'amplitude la plus forte et après on calcule sa fréquence par rapport à la phase qui correspond à cette amplitude. Comme le processus de la transformation spectrale dans Max est fait en temps réel, il y a besoin d'un fenêtrage. La détection de l'amplitude dominante est réalisée dans l'objet gen par une série de calculs.

3.3 Analyse spectrale

Dans la figure 3.3 on peut visualiser le processus de pitch tracking pour plusieurs harmoniques, le but est de révéler les composants d'un son et ainsi donc faire une analyse spectrale.

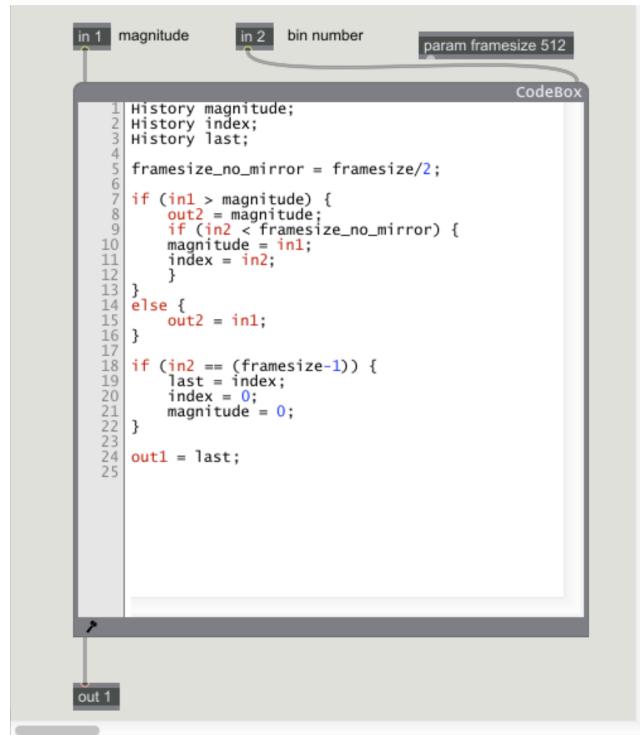


FIGURE 3.2 – Calcul de la magnitude Dominante dans un FFT

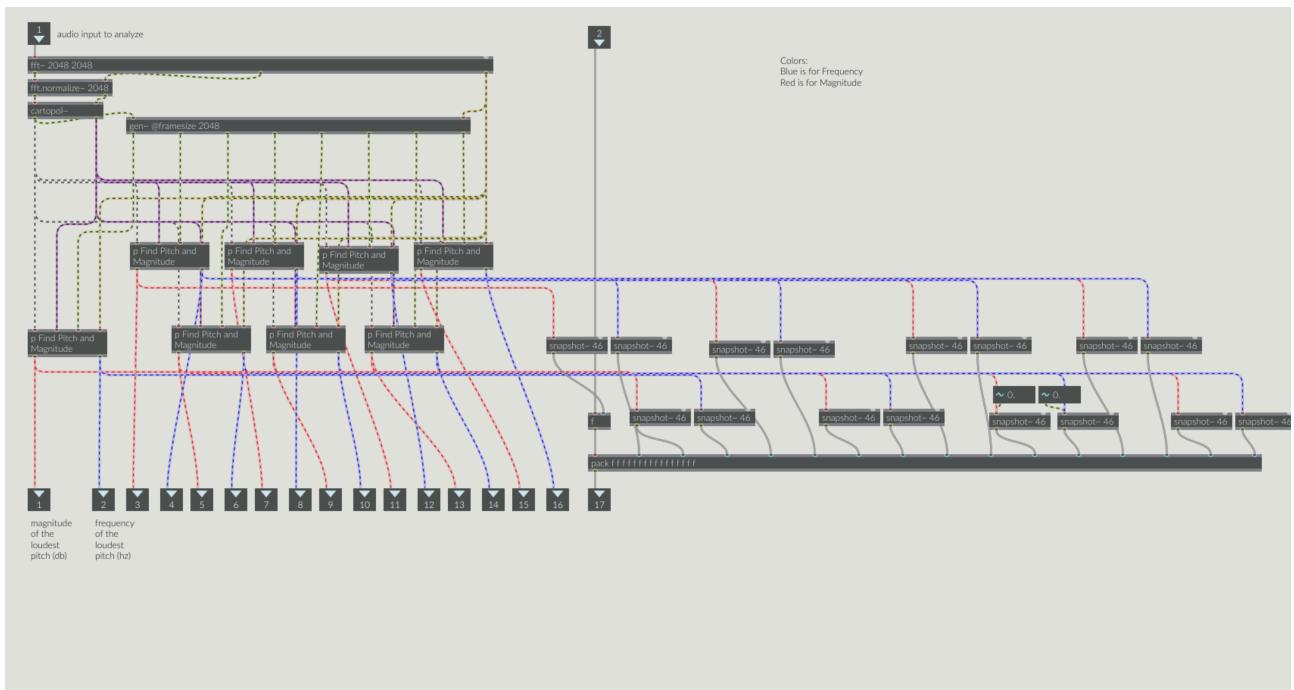


FIGURE 3.3 – Multiple Pitch tracking



FIGURE 3.4 – Calcule des Fréquences

Pour calculer plus de fréquences et par conséquent, pour concevoir le spectre d'un son, il existe un processus relativement simple. Dans la fonction gen , il y a le codebox de l'objet et son contenu. La copie de cet objet permet, l'extraction de fréquences harmoniques ou dominantes autant de fois que l'objet est copié. Comme mentionné ci-dessus, la première entrée de la zone de code est le volume d'échantillon courant, qui sera maintenant la sortie gauche de la zone de code précédente et a laissé le numéro d'échantillon. Ce processus est décrit graphiquement dans la figure 3.4.

3.4 Le vocodeur de phase

Pour accéder à une édition spectrale plus personnalisée et construire ainsi un propre vocodeur de phase, c'est important de avoir la pouvoir d'accéder à un fichier audio directement dans le sous-patch pfft . Cela signifie procéder à une FFT directe sur le son sans chercher un objet. Le raisonnement derrière cette logique repose sur la logique de base du vocodeur de phase pour l'étirement sonore et le changement de la hauteur. Le vocodeur nécessite une lecture indépendante du son à transformer pour chaque overlap FFT. Chaque image sonore de la lecture

doit être synchronisée avec son image sonore respective de la FFT. Par conséquent, une seule copie du son ne peut pas être lancée dans un objet `fftin`, mais plutôt dans un objet `fft`. L'objet `fft` exécute une FFT à spectre complet (c'est-à-dire en miroir), donc `fft` peut fonctionner en synchronisation avec les images FFT traitées dans l'objet `pfft` mais c'est nécessaire de faire quelques modifications sur l'objet `pfft` pour que il se comporte de la même manière.

Tout d'abord, l'objet `pfft` doit traiter des images sonore de la FFT à spectre complet, au lieu de l'image spectrale par défaut qui correspond à la moitié de la taille FFT (jusqu'à la moitié du Nyquist). Cela se fait facilement en ajoutant un cinquième argument non nul à l'objet `pfft`. Comme l'argument du spectre complet est le cinquième argument, nous devons fournir tous les autres arguments avant lui, y compris le quatrième argument, le début, qui sera défini sur la valeur par défaut de zéro.

Ensuite, parce que les objets `fftin` et `fftout` effectuent le calcul de la FFT à la phase zéro par rapport à la FFT (le premier échantillon de la fenêtre envoyée au FFT est le milieu de la fenêtre), et les `fft` et `ifft` les objets exécutent la FFT déphasée de 180 degrés, il faut assurer que tous les objets `fftin` et `fftout` du patch ont le même décalage de phase FFT utilisé dans les objets `fft`.

Cela peut être accompli en spécifiant un déphasage par rapport aux objets `fftin` et `fftout`. Une valeur de phase de 0,5 signifie un déphasage de 180 degrés, donc c'est la valeur préférable dans ce cas. Bien que `fftin` ne soit pas désiré, le `fftout` peut pratiquement être utilisé comme sortie pour l'objet `pfft`. Le fenêtrage automatique dans l'objet `fftout` devrait se comporter comme le fenêtrage manuel avec les objets `fft`.

Le buffer objet doit être accessible à deux endroits - à l'emplacement de la image sonore de la FFT actuelle et à l'emplacement de la image FFT précédente du buffer. C'est possible utiliser soit l'index ou l'objet `play` pour accéder au buffer. Et parce que la partie entrée du STFT est conçue manuellement, en utilisant l'objet `fft`, le signal lu dans le buffer doit être visualisé avant de l'envoyer aux objets suivants.

En utilisant deux objets `fft` distants d'un quart de l'overlap, deux overlap de la FFT peuvent

être calculées (l'image actuelle et la précédente). En utilisant cartopol pour convertir des coordonnées cartésiennes en coordonnées polaires, les amplitudes et les phases sont dérivées, puis il suffit de soustraire les phases de l'image sonore précédente de l'image actuelle pour obtenir des différences de phase pour chaque fenêtre. L'objet frameaccum peut être utilisé pour accumuler les différences de phase pour construire la phase d'exécution pour la sortie.

Sur notre vocodeur de phase on évitera d'utiliser le objet fftin .

3.5 Morphing spectrale

```

1 // autowatch 1;
2
3 // global variables
4 var s = 1;
5
6 function bang() {
7     if (myFunc == "Exponential")
8     {
9         x = Math.log(1.7182*x+1);
10        post("Exponential Interpolation");
11    }
12    else if (myFunc == "Logarithmic")
13    {
14        x = (x-1)*10;
15        x = Math.pow(2, x);
16        post("Logarithmic Interpolation");
17    }
18    else {
19        x = x;
20        post("Linear Interpolation");
21    }
22    outlet(0,x);
23}
24

```

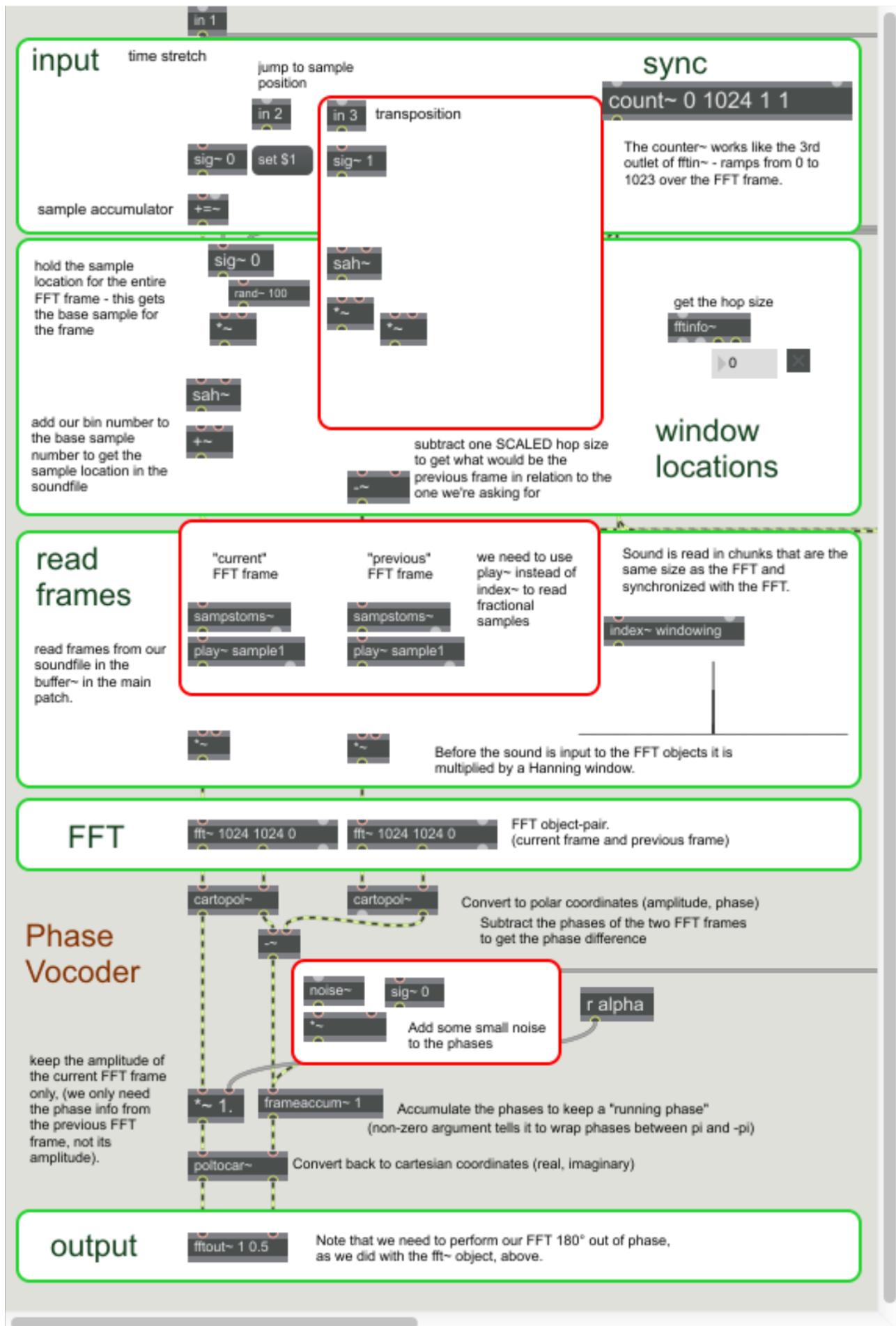


FIGURE 3.5 – Le vocodeur de phase

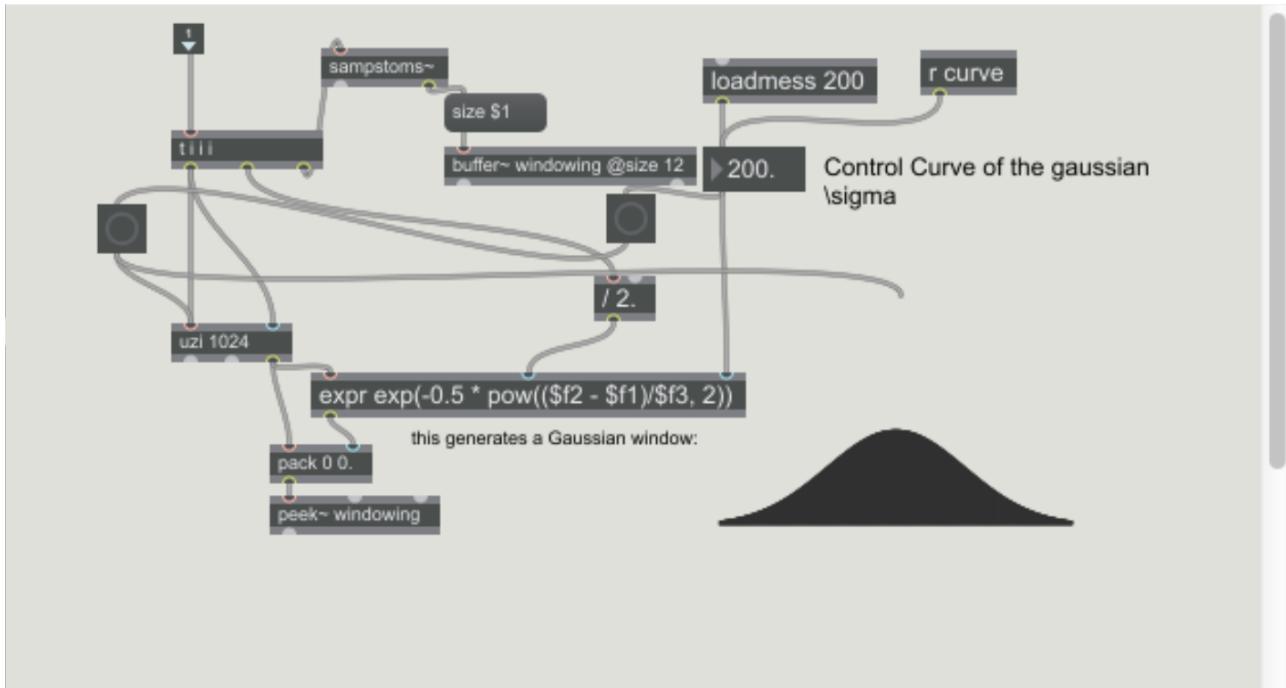


FIGURE 3.6 – Fenetrage gaussien

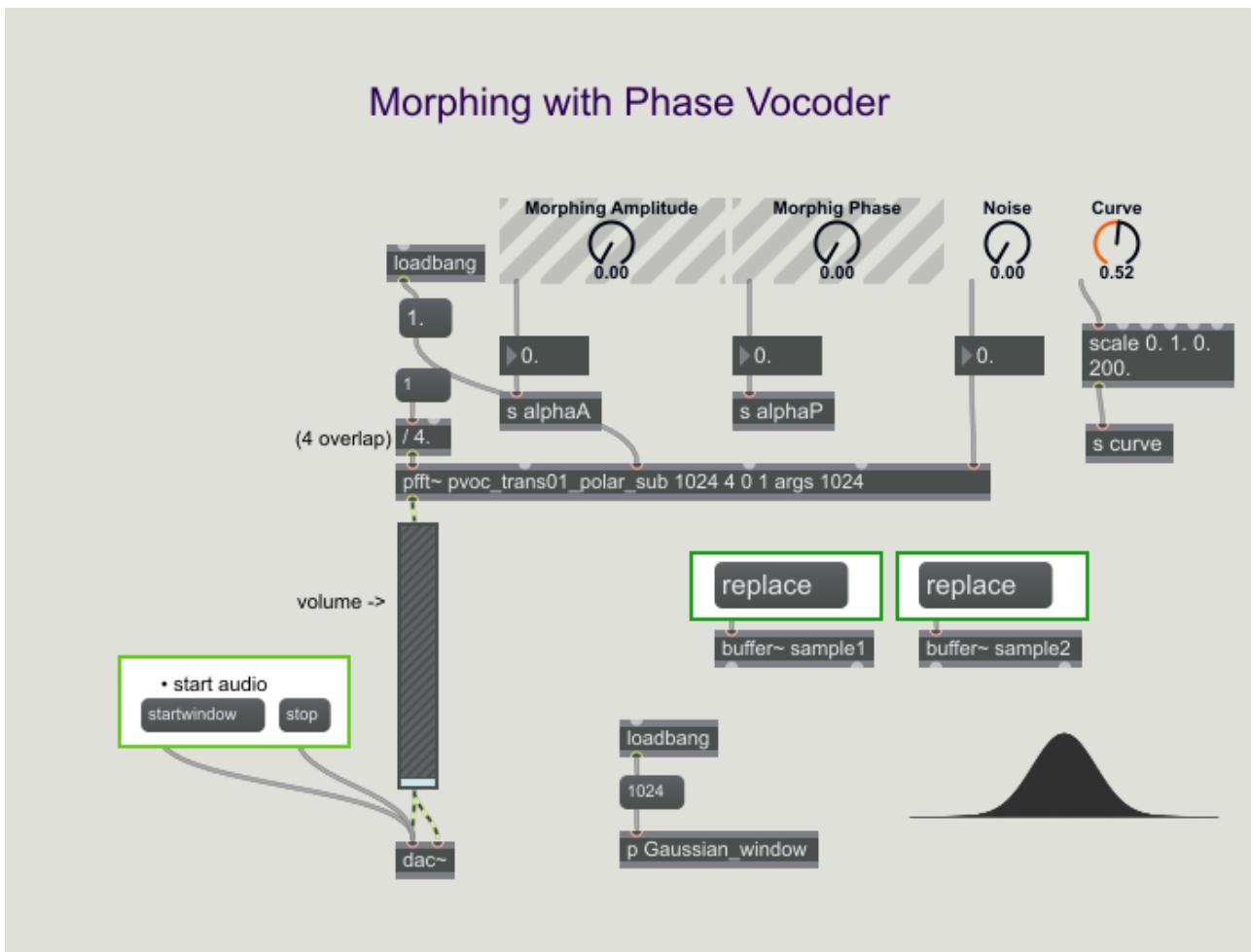


FIGURE 3.7 – Morphing en temps réel

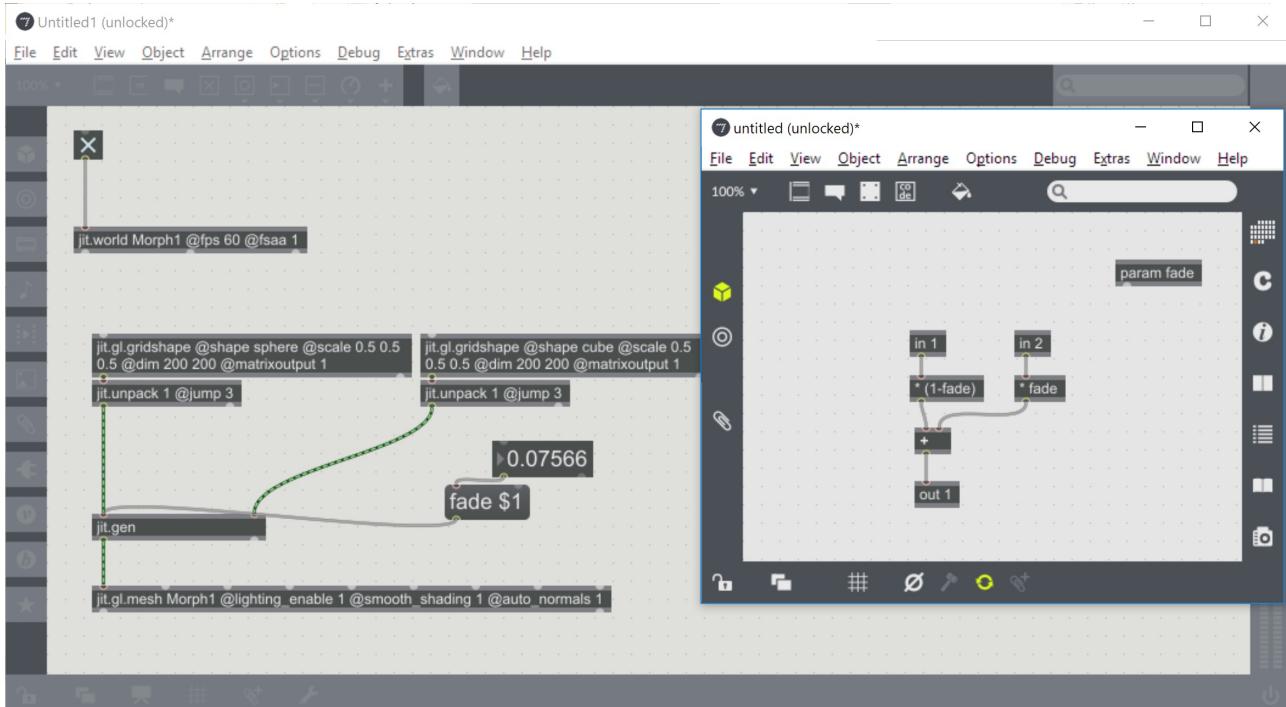


FIGURE 3.8 – Visual Morphing

```

25 function msg_float(v){
26     post("interpolation value " + v + "\n");
27     x = v;
28     bang();
29 }
30
31 function anything(){
32     var a = arrayfromargs(messagename, arguments);
33     post("received" + a + "\n");
34     myFunc = a;
35     bang();
36 }
```

3.6 Morphing visuel

En avançant sur le morphing visuel, j'ai implémenté le patch suivant avec l'aide du Jitter, pour une interpolation linéaire entre deux dessins.

En répétant la formule de base de morphing $M(\alpha) = \alpha\hat{S}_1 + [1 - \alpha]\hat{S}_2$ une patch sur le morphing

en 3D était implémenté avec l'aide de l'objet jit.gen (figure 3.8).

Donc, fondamentalement, un morphing visuel est facile à faire avec les fonctions 3D primordiales de Jitter telles que jit.gl.gridshape et jit.gen pour une personnalisation de la procédure de morphing. L'objet jit.gl.mesh est utilisé pour combiner le résultat du morphing alors que l'objet gen est contrôlé par un facteur de fondu.

À l'intérieur de gen, une procédure assez simple se produit. Les données multidimensionnelles provenant des matrices de localisation sont utilisées séparément pour chaque forme et leur amplitude est multipliée par le facteur a comme dans le morphing audio.

Bien entendu, nous pourrions également implémenter le script exponential.js pour une courbe de morphing différente sur le visuel.

Chapitre 4

Implémentations artistiques

4.1 *Introduction*

Cette section conclut un point de vue artistique sur les aspects abordés dans les chapitres précédents. À partir de la transformation de Fourier en vocodeur de phase et en passant au morphing par des interprétations artistiques des patchs techniques mis en œuvre.

L'objectif de ce chapitre est de soutenir l'idée que l'analyse et la re-synthèse spectrales, et donc en expansion, le développement du signal musical sont utiles aux scientifiques et aux artistes. En effet, les détails techniques antérieurs sont utilisés à des fins artistiques et esthétiques. Les exemples suivants consistent en une expérimentation personnelle et une interprétation du matériel technique déjà présenté.

4.2 Le vocodeur de phase - *Une capacité sans fin*

Dans le but de combiner plusieurs techniques du vocodeur de phase, nous avons implémenté un «*super* vocodeur de phase». Cet outil spectral comprend l'étirement temporel, le freeze temporel, le décalage de hauteur, le morphing et d'autres effets bénéficiant de la fonction élémentaire du vocodeur de phase.

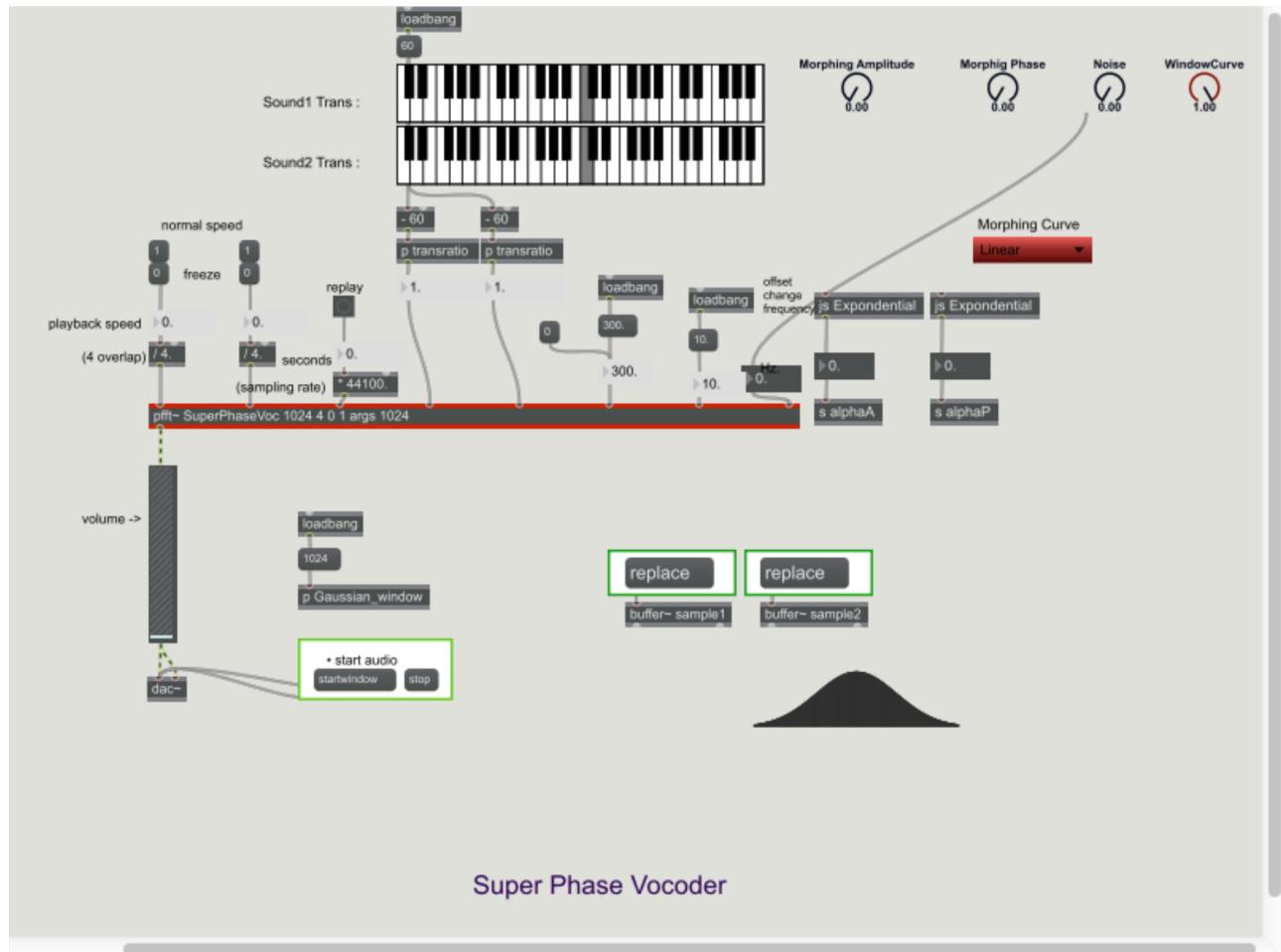


FIGURE 4.1 – Super Phase Vocoder I

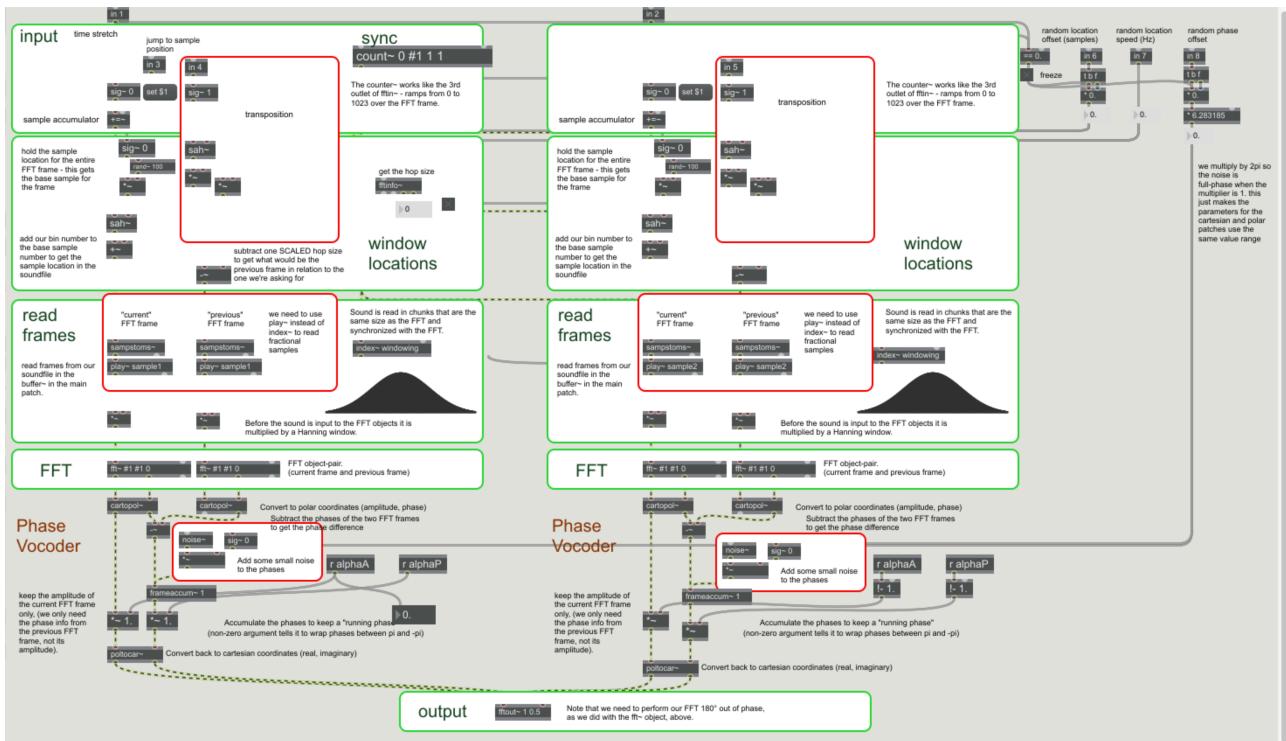


FIGURE 4.2 – Super Phase Vocoder II

Chapitre 5

Conclusion

5.1 Résumé de la recherche

Cette recherche sert à des fins documentaires, le but est d'informer les musiciens et les compositeurs des notions complexes. Ce travail me sera aussi enrichissant au niveau personnel, il me permettra d'approfondir mes connaissances dans ce domaine. Le spectre est une terminologie difficile à comprendre pour les musiciens et un point de vue plus musical va faciliter sa compréhension. En approfondissant sur le terrain du morphing, on découvrira de nouvelles manières pour manipuler cet effet. Cette problématique, constituera le but final de la recherche.

5.2 Applications

Les applications du vocodeur de phase sont quasiment infinies, comme indiqué dans le chapitre sur l'implementation. Quelques propositions suivront pour afficher les capacités d'analyse spectrale et le vocodeur de phase.

Le vocodeur de phase peut être utilisé pour analyser la voix et la transformer en texte. Les pics et les fréquences peuvent être déduits de certaines voyelles et consonnes pour prédire les lettres exactes utilisées et produire la forme écrite du son.

Le vocodeur de phase humain par voix peut également transformer un certain type de voix, par exemple la voix d'un homme, en une femme ou tout autre type de voix en se fondant entre les caractéristiques qui déterminent une voix masculine, féminine ou autre.

Les outils spectraux et spécialement ceux d'analyse-synthèse sont fréquemment utilisés par les compositeurs via des programmes DAW. Une bibliothèque célèbre serait TRAX par Ircam tools.

5.3 Recherche pour l'avenir

De plus, je propose une suite de ma recherche sur les processus spectraux en utilisant les principes tels que Deep Learning. Spécifiquement un approche sur le Morphing spectrale en temps réel connecte aux networks neurals¹.

1. Jesse Engel. Making a neural synthesizer instrument, 2008.

Listings

Exponential.js	27
--------------------------	----

Bibliographie

Steven W. Smith. The scientist and engineer's guide to digital signal processing. *California Technical Publishing*, 1997.

Hermann L F. Helmholtz. *The Sensations of Tone*. New York, 1895.

Tadej Droljc. *STFT Analysis Driven Sonographic Sound Processing in Real-Time using Max/MSP and Jitter*. 2011.

Jean-François Charles. A tutorial on spectral sound processing using maxmsp and jitter. *Computer Music Journal*, pages 87–102, Printemps 2008.

Alan V. Oppenheim et Ronald W. Schafer. Discrete-time signal processing. *Prentice Hall Press*.

Richard Ducas et Cort Lippe. The phase vocoder - part i, 2 novembre 2006. URL <https://cycling74.com/tutorials/the-phase-vocoder-%E2%80%93-part-i>.

Richard Ducas et Cort Lippe. The phase vocoder - part ii, 2 juillet 2007. URL <https://cycling74.com/tutorials/the-phase-vocoder-part-ii>.

Axel Roebel. Morphing sound attractors. In *3rd. World Multiconference on Systemics, Cybernetics and Informatics (SCI'99) and the 5th. Int'l Conference on Information Systems Analysis and Synthesis (ISAS'99)*, 1999, Florida, United States, Proc. of the 3rd. World Multiconference on Systemics, Cybernetics and Informatics (SCI'99), 5th. Int'l Conference on Information Systems Analysis and Synthesis (ISAS'99), 1999.

Fernando Villavicencio et Xavier Rodet Axel Roebel. On cesptral and all-pole based spectral

envelope modeling with unknown model order. *Pattern Recognition Letters*, (28) :1343–1350,

2007.

Mark Dolson. The phase vocoder : a tutorial. *Computer Music Journal*, pages 14–27, Printemps 1986.

William Fulton. Introduction to intersection theory in algebraic geometry. In *Regional Conference Series in Mathematics*, number 54, 1983.

Axel Roebel and Xavier Rodet. Efficient Spectral Envelope Estimation and its application to pitch shifting and envelope preservation. In *International Conference on Digital Audio Effects*, pages 30–35, Madrid, Spain, September 2005. URL <https://hal.archives-ouvertes.fr/hal-01161334>. cote interne IRCAM : Roebel05b.

Mark Goresky et Robert MacPherson. On the topology of complex algebraic maps. In *Algebraic Geometry Proceedings, La Rábida, Lecture Notes in Mathematics*, number 961, 1981.

Axel Röbel. Morphing Sound Attractors. In *3rd. World Multiconference on Systemics, Cybernetics and Informatics (SCI'99) and the 5th. Int'l Conference on Information Systems Analysis and Synthesis (ISAS'99)*, Florida, United States, 1999. URL <https://hal.archives-ouvertes.fr/hal-01253228>.

Aldo Piccialli Aldo Giovanni De Poli Curtis Roads, Stephen Travis Pope. *Musical Signal Processing*. London et New York, 1997.

J. Fourier and A. Freeman. *The Analytical Theory of Heat*. Kessinger Publishing, 2010. ISBN 9781164439561.

Michael Kateley Klingbeil. *Spectral Analysis, Editing and Resynthesis : Methods and Applications*. Université de Columbia, 2009.

Curtis Roads. *The Computer Music Tutorial*. MIT Press, Cambridge, MA, USA, 1996. ISBN 0262680823.

C. Roads, S.T. Pope, A. Piccialli, and G. De Poli. *Musical Signal Processing*. Taylor & Francis, 2013. ISBN 9781134379705.

Gérald Grisey. Écrits : ou l'invention de la musique spectrale. MF Éditions, 2008.

Javier R. Movellan. A tutorial on gabor filters, 2008. URL <http://mplab.ucsd.edu/tutorials/gabor.pdf>.

Jesse Engel. Making a neural synthesizer instrument, 2008. URL <https://magenta.tensorflow.org/nsynth-instrument>.

Joachim Heinz et Andes Cabrerra et Alex Hofmann et Iain McCurdy et Alexandre Abrioux. Fourier transformation / spectral processing.

Emmanouil-Nikolaos Karystinaios. An investigation into the spectral music idiom and timbral analysis functionality with max smp jitter. Master's thesis, Septembre 2017. preprint.

Cavin Wu. How computer technology influences art and design programs in higher education. Master's thesis, La Sierra University, 2006.

Jonathan Boley. *Auditory Component analysis using perceptual pattern recognition to identify and extract independent components from an auditory scene*. PhD thesis, Université de Miami, 2005.

Johannes Grünwald. Theory, implementation and evaluation of the digital phase vocoder in the context of audio effects, 2010.

Daniel W. Griffin et Jae S. Lim. Signal estimation from modified short-time fourier transform. *IEEE Transaction on acoustics, speech, and signal processing*, ISSE Vol. 32 :236–243, Avril 1984.

Marcelo Freitas Caetano and Xavier Rodet. AUTOMATIC TIMBRAL MORPHING OF MUSICAL INSTRUMENT SOUNDS BY HIGH-LEVEL DESCRIPTORS. In *International Computer Music Conference*, pages 11–21, United States, June 2010. URL <https://hal.archives-ouvertes.fr/hal-00604390>.

Jown Strawn. Introduction to digital sound synthesis. *Digital Audio Engineering*, pages 141–134, 1985.

Richard Kronalnd-Martinet et Thierry Voinier et Solvi Ystad et Kristoffer Jensen. Computer music modeling and retrieval. *4th international Symposium CMMR*, 2007.

Joshua Fineberg. Guide to the basic concepts and techniques of spectral music. *Contemporary Music Review*, pages 81–113, 2000.

Dennis Gabor. Theory of communication. *ICMC*, 1944.

Anaik Olivero, Bruno Torrésani, Philippe Depalle, and Richard Kronland-Martinet. Sound morphing strategies based on alterations of time-frequency representations by Gabor multipliers. In *AES 45th International Conference on Applications of Time-Frequency Processing in Audio*, page 17, Helsinki, Finland, March 2012. URL <https://hal.archives-ouvertes.fr/hal-00682959>.

Axel Roebel. A new approach to transient processing in the phase vocoder. In *6th International Conference on Digital Audio Effects (DAFx)*, pages 344–349, London, United Kingdom, September 2003a. URL <https://hal.archives-ouvertes.fr/hal-01161124>. cote interne IRCAM : Roebel03a.

Axel Roebel. Transient detection and preservation in the phase vocoder. In *International Computer Music Conference (ICMC)*, pages 247–250, Singapore, Singapore, October 2003b. URL <https://hal.archives-ouvertes.fr/hal-01161125>. cote interne IRCAM : Roebel03b.

Geoffroy Peeters. A large set of audio features for sound description in the cuidado project. *A large set of Audio Features for Sound Description*, 2004.

Anne Sédes. Spectralisme français, du domaine fréquentiel au domaine temporel, analyse, modélisation, synthèse. . . et prospectives. *The Foundations of Contemporary Composing*, 2002.