Projet Son

Benjamin Haté Thibaud Priou

Février 2019

1 Introduction

Dans le cadre du module de Synthèse Sonore, nous avons été invité à créer un code d'analyse de sons en s'inspirant des exercices que nous avons eu pendant l'année.

Parmis les possibilités d'implémentation, nous avons décidé de faire une analyse sonore d'un son au format *.wav*. Cette analyse comprent :

- Une analyse de tempo qui nous permet de connaîte les battements par minutes (*bpm*) du son en entrée.
- Une analyse du pitch global qui nous permet de connaître la tonalité du son en entrée.
- Une analyse de l'autosimilarité qui nous permet de connaître des motifs récurrents du son (e.g : les refrains des chansons).

2 Explication du code

Dans cette partie, nous allons expliquer les points de code utilisés pour analyser le son en entrée : le tempo, la tonalité et l'autosimilarité.

2.1 Matrice d'auto-similarité

Sur la Figure 1, on peut voir la matrice d'auto-similarité d'un son au format *.wav*. Cette matrice montre les motifs récurrents dans le son. Ces motifs sont visibles par le croisement des valeurs fortes (les plus sombres).

FIGURE 1 – Matrice d'auto-similarité d'un son au format *.wav*

2.2 Analyses spectrales

Afin de réaliser les divers traitements sur le morceau, il est nécessaire de réaliser plusieurs Transformations de Fourrier (TF) :

— pour calculer le flux spectral (tempo)

— pour calculer des chromas pour matrice d'autosimilartié

— pour calculer des chromas pour détecter le pitch

Alors que le calcul de flux spectral a besoin que la TF soit réalisée sur des petites fenêtres, la détection de pitch nécessite d'avoir une fenêtre suffisamment grande pour différencier plusieurs notes, surtout dans les basses fréquences. Pour se faire, nous avons calculé deux FFT différentes :

— une FFT sur 1024 échantillons temporels est réalisée pour le tempo et la matrice d'autosimilarité

— une FFT sur 32768 échantillons temporels est réalisée pour le pitch.

Ainsi, pour le pitch, chaque échantillon de la FFT correspond à une bande fréquentielle de "*sample_rate / 32768*" soit environ 1.35Hz pour un taux d'échantillonage de 44100Hz. Il est à noter qu'il nous était également possible de réaliser une transformation de Fourrier non-uniforme, car les fréquences des notes aiguës sont plus espacées et ne nécessite pas un échantillonage aussi précis que les

notes basses fréquences.

2.3 Compilation et exécution

Pour créer les fichiers exécutables, il faut se placer dans le dossier du code et lancer la commande `make`.

Pour lancer le code, il faut lancer la commande `./audioAnalyzer <input_file> <output_file>` où `<input_file>` correspond à un fichier au format *.wav* et `<output_file>` est le chemin de l'image PPM contenant la matrice d'autosimilarité.

3 Conclusion

Ce projet nous a appris à utiliser les outils que nous avons vu en cours afin d'analyser un morceau musical.

Les analyses que nous avons exploité permettent de connaître les propriétés essentiels du son, comme son tempo. La cible de notre algorithme est la musique et les chansons mais il fonctionne aussi avec un son quelconque. Cependant, les informations données par la matrice d'auto-similarité sont plus intéressantes lorsque des chansons sont traitées.