Rapport:
Traitement du signal temps réel
2020/2021

Réalisé par:
Rachidi Lalaoui Omar
Guelbi Mahdi

Encadré par:
Mr. Thomas Hueber
Mr. Laurent Girin

introduction:

Le but de ce projet est d'effectuer un effet de reverb en temps réel sur un signal audio input. Cela est effectué en faisant la convolution entre le signal input et un signal "impulse" représentant un echo. Afin de créer le projet nous avons travailler sous un environnement linux en utilisant la bibliothèque rtaudio. Le langage de programmation utilisé est le C++, c'est un langage proche de la machine et qui offre un grand degré de liberté en terme d'allocation de mémoire et de calcul, ce qui permet de créer des codes rapides et legers.

Réalisation:

L'organisation du code se divise en 3 parties:
- une classe "ArgumentHolder": elle gère les paramètres du système.

```
string channelsArg = getCmdOption(argc, argv, "-ic=");
string sampleRateArg = getCmdOption(argc, argv, "-fs=");
string iDeviceArg = getCmdOption(argc, argv, "-input=");
string oDeviceArg = getCmdOption(argc, argv, "-output=");
string convTypeArg = getCmdOption(argc, argv, "-conv=");
string oChannelsArg = getCmdOption(argc, argv, "-oc=");
string bufferFramesArg = getCmdOption(argc, argv, "-bs=");
string inputFileArg = getCmdOption(argc, argv, "-file=");
string mArg = getCmdOption(argc, argv, "-m=");
```

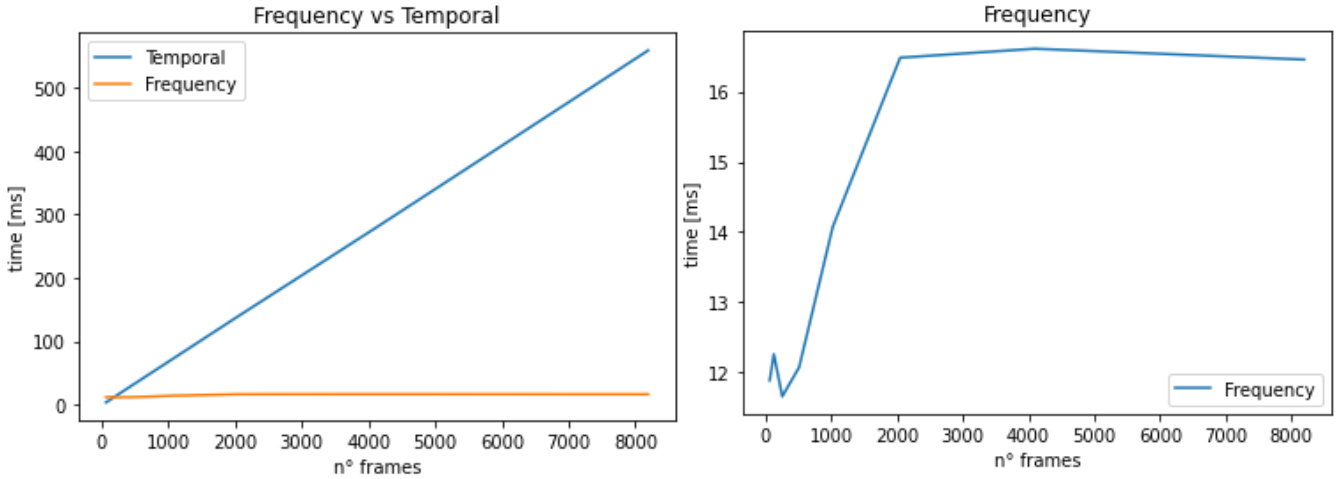
- une classe "AudioHandler": cette classe contient toutes les variables et fonctions en relation avec la gestion de l'audio (buffer, fonctions d'initialisation, de convolution et d'overlap add).
- le fichier principal: qui regroupe le tout en contenant la fonction "callback", l'intialisation et l'ouverture du stream adac...etc.

On a implémenté les deux sortes de convolutions à savoir **temporelle** et **fréquentielle**. Afin d'avoir un traitement en temps réel nous avons utilisé la méthode "overlap-add" qui est une méthode d'ajout par chevauchement utilisée pour diviser le flux et les opérations du signal en petits segments. On utilise donc cette méthode avec la convolution et la transformée de Fourier rapide, permettant aux signaux d'être convolués en multipliant leurs spectres de fréquence.

Comparaison temporelle vs fréquentielle:

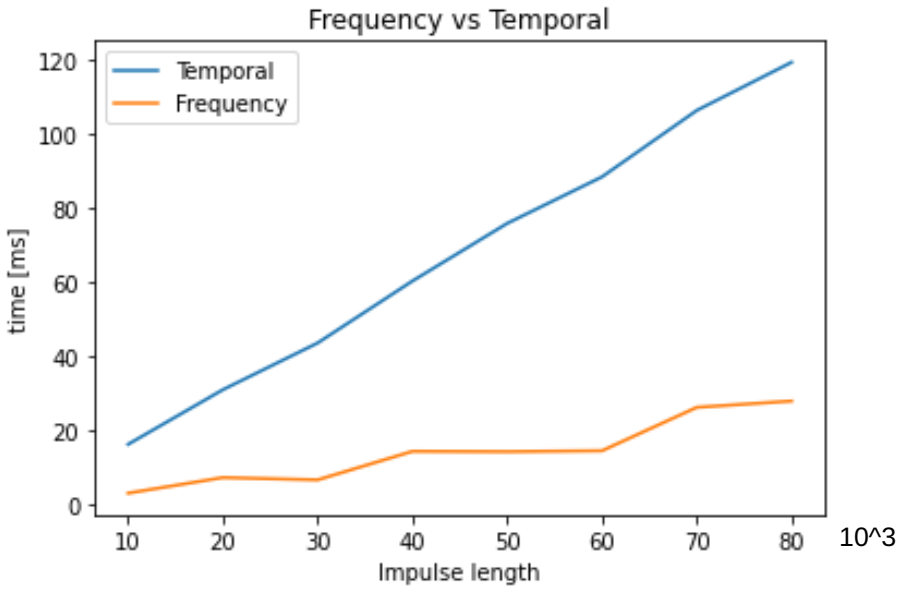
Afin d'évaluer les deux méthodes (temporelle, fréquentielle) en terme de temps de latence, nous avons lancé plusieurs tests suivant deux paramètres: nombre de frame et taille de l'impulsion.

Temporelle vs Fréquentielle : N°Frames



On remarque que comparé à la convolution temporelle le temps de latence de la convolution fréquentielle ne change quasiment pas en fonction du nombre de frames du buffer input (évolution logarithmique). Alors que la convolution temporelle augmente de manière linéaire. Donc, la convolution fréquentielle est beaucoup plus efficace et ceci apparaît clairement lors du lancement du système aussi.

Temporelle vs Fréquentielle : Taille filtre



On remarque là aussi qu'en augmentant la taille du filtre impulsionnel convolué avec le signal audio d'entré, la latence augmente. Et on remarque aussi la même différence entre la convolution temporelle et fréquentielle. En effet pour une taille de l'impulse grande la latence de la convolution temporelle augmente d'une manière linéaire alors que la convolution fréquentielle augmente d'une manière logarithmique.

Bien sûr il faut trouver un compromis entre la latence et les 2 autres paramètres et qui sont la taille de l'impulsation et le nombre de frame qu'on utilise. Une bonne configuration qui peut être considérée est:
- convolution fréquentielle.
- N frames: 512 ou 1024.
- Taille: 35k.

Conclusion

En conclusion nous la performance d'un système temps réel dépend du type d'opération qu'on utilise pour avoir nos résultats. En effet, nous avons constaté durant ce projet qu'il y a une très grande différence entre le fait d'utiliser une convolution temporelle et fréquentielle ce qui impacte grandement la performance de nos systèmes temps réels. En effet en utilisant une convolution fréquentielle qui a une complexité logarithmique on diminue considérablement le temps d'exécution de chaque opération.