Plan de projet

Par

François Poitras

Présenté à

Olivier Bélanger

Dans le cadre du cours de

Création Musicale en Language Python 2 (MUS3327)

22 avril 2017

Université de Montréal

1 Plan de travail

1.1 Énoncé

Construction en temps réel de progressions d'accords. Le but de programme est d'aider l'utilisateur à comprendre les effets de changements d'accords sur une mélodie générée de manière aléatoire.

En raison du manque de temps et de la complexité demandée par la génération de mélodie à été mise de coté.

1.2 Analyse des besoins

L'interface graphique aura deux zones principales, la première étant une représentation de la progression d'accord actuelle. La seconde zone contiendra les accords que l'utilisateur peut remplacer. Pour faire le remplacement, il utilisera un double clic. Si le temps le permet, une méthode de "Drag and drop" sera implémentée. Si le temps ne le permet pas, l'utilisateur aura accès à une zone de texte ou le programme pourra *parser* le contenu pour en déduire la progression à utiliser. Par exemple, l'utilisateur pourra entrer "Dm7 - G7 - Cmaj7" dans la zone de texte. La plupart des contrôles se feront avec la souris et la saisie de texte sera avec le clavier.

Pour faciliter la visualisation de l'effet des changements d'accord, l'accord en cours de lecture (voir section "méthodes" pour plus de détails) sera mis en évidence avec un changement de couleur.

1.3 Acquisition de connaissances

Une recherche sera effectuée sur la modélisation d'instruments afin d'obtenir un rendu plus réaliste. L'application tentera d'émuler un piano, il y aura donc une recherche dans ce sens. Au besoin, j'utiliserai l'enregistrement d'un vrai piano pour avoir une idée plus précise du son qu'il faut représenter. Aussi, étant donné le très large intervalle couvert par un piano, il se peut qu'un autre instrument de basse soit nécessaire, dans le cas où la synthèse ne soit pas assez bonne dans tout le spectre couvert. Le plus simple est probablement un instrument à vent, comme une clarinette basse. Si le temps le permet, l'application implémentera aussi des instruments à cordes, comme une contrebasse ou une guitare. À première vue, la synthèse de ces instruments est plus complexe et c'est pour cette raison qu'ils ne seront pas prioritaires.

En raison de ce manque de connaissances, il est difficile pour l'instant de prévoir comment va se dérouler la modélisation, car j'ai l'impression qu'en plus de la recherche, il y aura une phase d'essai-erreur pour explorer Pyo et déterminer quels générateurs de signaux combinés à quels effets de la librairie peuvent être utilisés ensemble pour modéliser quelques instruments.

Pour les questions de construction d'accords ou, de manière plus générale, il faudra des recherches pour connaitre les règles. Pour simplifier le projet, l'application va limiter le nombre d'accords disponibles. Il ne sera pas possible, dans les plans initiaux, de créer n'importe quel accord et l'utilisateur sera limité à une banque pré-définie.

1.4 Modèle

Un instrument à vent produit un son grâce aux vibrations de l'air qui résonnent dans un tube de longueur variable. Dans ce cas, il n'y a pas d'excitation intiale et le son cesse dès que le joueur cesse de souffler. L'attaque de l'instrument est ainsi beaucoup plus longue et le *sustain* dure aussi longtemps que le musicien souffle.

Le piano est quant à lui un coup de marteau sur une corde de métal qui résonne par la suite. Il faudra donc simuler ce coup, qui est une excitation initiale, puis la résonnance qui s'en suit. Notons que dans les deux cas, nous n'avons pas une seule fréquence de résonnance, mais plusieurs. On notera la présence d'harmoniques de la fréquence fondamentale.

Il m'est malheureusement difficile de décrire plus en détail la modélisation musicale, car mes connaissances en synthèse audio sont plutôt limités et il me faudra explorer les processus audio de Pyo plus en profondeur pour avoir une idée plus précise de ce qu'il faut comme travail pour simuler de façon adéquate un piano et un instrument à vent.

1.5 Méthodes

Une classe sera dédiée à contenir un "dictionnaire" de gammes et d'accords. Afin de faciliter l'abstraction, toutes les définitions seront basées sur la première octave MIDI. Par exemple, on pourra définir un accord majeur comme étant l'ensemble des notes $\{0, 4, 7\}$. On représentera une gamme majeur de façon similaire, comme étant l'ensemble $\{0, 2, 4, 5, 7, 9, 11\}$. Pour que le programme puisse distinguer les deux cas, les accords et les gammes seront dans des dictionnaires séparés. La classe permettra de transposer — par multiples de douze pour l'octave et en additionnant ou soustrayant plus ou moins 11 pour la tonalité — au besoin de l'utilisateur. Cette classe n'implémentera pas à proprement parler de processus audio, mais les classes utilisant de l'audio pourront utiliser ce dictionnaire comme référence.

Une autre classe aura le rôle de gérer l'interface graphique. Celle-ci sera composée d'une zone qui contiendra la progression d'accord (à la manière d'une grille d'accord). La version par défaut de la grille aura la forme d'un accord par mesure. Donc, si la progression dure 4 accords, la grille aura quatre cases. Il sera possible d'en ajouter ou d'en supprimer à l'aide d'un bouton. Puisque le but de l'application est d'entendre l'effet de changement d'accords, le programme va mettre en évidence l'accord courant, avec une couleur de texte différente.

Une classe sera responsable d'implémenter différents instruments. Pour simplifier les choses, une classe mère aura les composantes principales et les classes enfants utiliseront l'héritage pour implémenter leur instrument. La classe mère aura entre autres des propriétés de volume, d'orientation sonore (gauche/droite) et de tempo. Les classes enfant utiliseront des chaînes de processus audio pour arriver à simuler leur instruments respectifs. En principe, uniquement les classes d'instruments vont utiliser les différents modules de la librairie sonore. Pour initialiser un instrument, il faudra lui passer un mode de fonctionnement, soit "mélodie" ou "accord". Dans le premier cas, il faudra aussi préciser quelles notes on désire utiliser. La série de note sera utilisée de façon aléatoire pour générer une mélodie, avec un tempo déterminé. Dans le deuxième cas, un accord sera construit à partir du dictionnaire, une fois par temps.

1.6 Résumé du développement

1.6.1 Problèmes rencontrés

1.7 Perspectives