HyungSeok Yoon

Professor Hoerning

2019/03/26

Music and Engineering

**Homework 3 Write Up**

This document serves as an opportunity for me to explain some of the aspects of the code that I have written, and other things that I did.

1. **hw3.m**

This file serves as an input variable midi file is to be read.

1. **midiparse.m**

This function takes in the name of the midi file as the input, and outputs miscellaneous information such as MIDI file format, number of tracks, and ppqn, as well as track information as track\_info cells. The track\_info cells consist of all the applicable information that is going to be helpful in sound synthesis. It stores time stamps, frequencies, and velocities of all the notes on and notes offs as well as volume, patch number, and name for each tracks.

1. **soundsynth.m**

soundsynth function takes in outputs of midiparse function, and forms a sound output. It extracts each note from the track\_info cells and create sound, and mix them together at the end to create an overall sound.

1. **createsound.m**

createsound function creates a sound of certain length, frequency, and style depending on the patch number. I only put in my additive synthesis bell, clarinet, and sinewave with attach and decay to simulate velocity because I don’t have that many synthesis bank.

1. **b2dec.m**

b2dec is a small function that I used to convert a string of decimal numbers to one decimal number with hexadecimal rules of combining. For example [1,0] will turn into 256\*1 + 0 = 256.

1. **inv\_VLQ.m**

inv\_VLQ is a function that I used to take care of delta times that are two string long. It translates variable length quantity into correct decimal. It only works for two bytes cases.

1. **wave\_shape.m**

This is the function is demonstrating my clarinet function generated in the previous hw2.

1. **attack\_decay.m**

I created this function similar to HW1 to demonstrate the attack and decay implementation. This could make my output to be seen clearer.

**Performances:**

I have tested my code with all the sample midi files provided by professor and all sounded as what they are should be. The distinction of each tracks was not very noticeable, but I could see the differences.

Mario.mid file sounds good, I think the reason might be because the original sound also is generated by machine and it has all 8-bits.

ROW.mid has inharmonious sounds, but you can still hear that it is playing what it is supposed to play.

Furelise.mid sounds wrong at the first time. I think the piece I was converting should not be heard in that way. Some of the notes should have been appeared more times compared to what I was listening from the Matlab. However, after I listened to other people’s sound output, I think there can be a possibility that my code is right in some ways. If it is not, I couldn’t understand what was causing the problem, but one of the possible problems can be the tempo changes with the time. I thought tempo was constant for all midi files, but furelise.mid file was storing tempo changes in the first track that applies to the notes track. I currently have a single constant value of tempo in my code that applies to all the tracks. If I created different arrays for each tempo, then track playing should have been improved.