MUSI 6202 Spring 2021

Final Project Overview

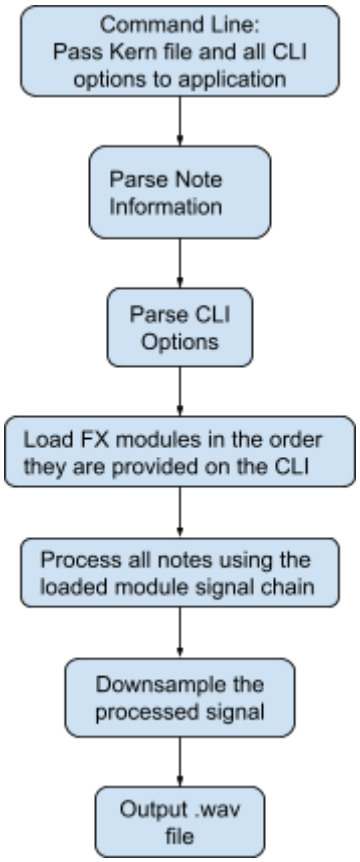
Group 4: Virgil Breeden, Rhythm Jain, and Laney Light

Code

Our code is accessible on github in the following location:
https://github.com/vbreeden/music6202/tree/main/FinalProject

Overview

We developed a python-based synthesizer that can be run using the command line. Our process uses symbolic music notation (Kern files) as input, and we have created several example melodies in this format for testing purposes. The system produces output files in .wav format, downloaded to the user's local machine. The data flow is illustrated in the figure below. Example code with different combinations of effects is provided at the end of this document.



Python libraries

To run our code, the following Python modules need to be installed on the user's local machine: via 'pip install' :

1. soundfile
2. wavio
3. Music21
4. alive_progress

Components

Synth Engines

1. Additive
2. Wavetable

Command Line Interface (CLI)

- Syntax for the command line options can be found by running the following command:
  python finalsynth.py -h
- Parameters for each modules can be adjusted in the CLI
- Detailed information about allowable values for each of the parameters can be found in readme.md

Serial signal chain / signal pipeline

- Implementation is modular
- Effects/filter are applied in the order specified by the user

Modulated Effects

- Chorus
- Vibrato
- Delay line with feedback

Convolution Effect

- FIR Reverb
  - Several impulse response WAV file are provided as options for the convolution

Filters

- Low pass
- Band pass

Input data

- Four example Kern files are provided for the user to choose from, each containing a single melody line (monophonic)
- Process input using music21: 3rd party python library
- For each note, get pitch, amplitude, note on/off times

Audio output

- soundfile 3rd party library
- File output in .wav format
- Configurable output sample rate and bit depth via CLI
    - Use 48k internally
    - Discrete options to select from for the sample rate & bit depth
    - Downsample using cubic interpolation and dither

References

In addition to the jupyter notebooks provided in the class resources, we used the following open source synths for reference -

1. PySynth - https://mdoege.github.io/PySynth/
2. Yoshimi - http://yoshimi.sourceforge.net/
3. Amsynth - http://amsynth.github.io/
4. Dexed - http://asb2m10.github.io/dexed/