6.111 Project Proposal

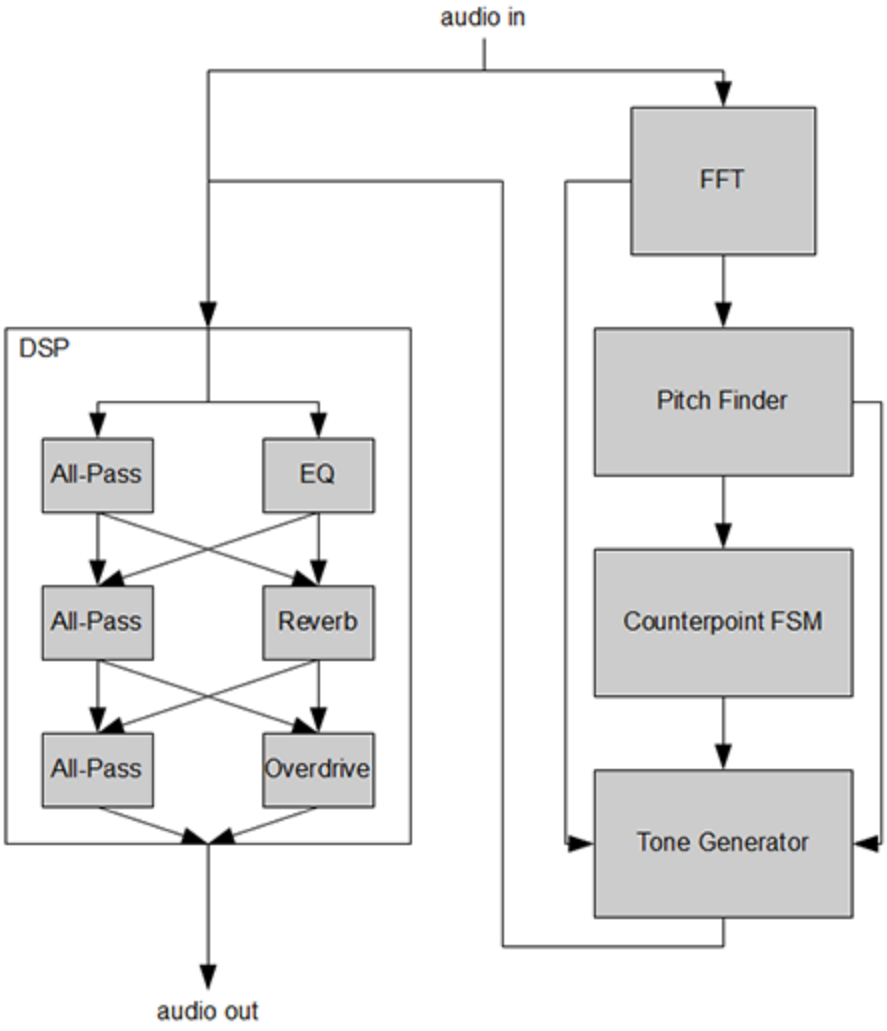
Real Time Harmonizer and Mixer

Elizabeth Hass Christopher Chin

November 9, 2010

Project Description

We propose to build a system that is capable of creating a modulated version of the incoming audio signal as a counterpoint melody in real time. We also propose to build in a variety of effect filters to change the sound of the produced counterpoint in real time and to play with both the original audio and counterpoint in a post processing mode.



Module Descriptions

Fast Fourier Transform Module (Ellie)

Our project requires an excellent pipelined fast fourier transform module. We hope that the included FFT with the 6.111 labkit meets our needs. If it does not, we plan to draw on the FFT code from previous years. Both the PerfectPitch project from 2007 and the iSing Voice Harmonizer from 2009 implement FFTs meet our needs in a post-processed implementation of our design. We hope to complete a real-time harmonizer, but as a fall back a reasonable playback of a recording with a generated counterpoint and effects would still be interesting.

Pitch Finder Module (Ellie)

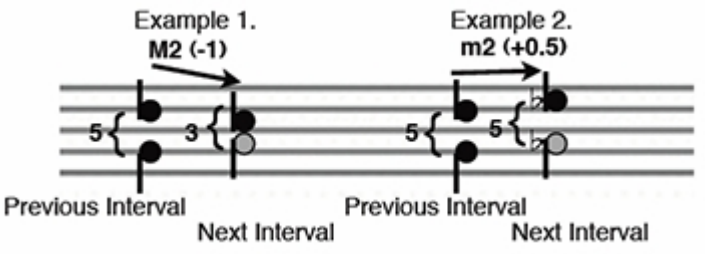
The pitch finder, probably based heavily on the 2007 PerfectPitch module, will take the stored FFT values and determine the note played by finding the peak with the highest magnitude and translating the corresponding frequency to the nearest discrete note value C2(65 Hz) - C6(1046 Hz). That note value will in turn be output to the counterpoint finite state machine.

Counterpoint Finite State Machine Module (Ellie)

Input: Note value for current input, C2(65 Hz) - C6(1046 Hz)
Stored values: Last Interval
Output: New Interval

The counterpoint module will attempt to create a melodic line that follows 18th century first species counterpoint rules with respect to the incoming notes determined by the pitch finder. The basic rules of first species counterpoint that I intend to follow are:

1. No interval larger than a tenth will be played
2. Allowed intervals are limited to the major third, perfect fourth, perfect fifth, major sixth, perfect eight, and major tenth.
3. Contrary motion between the melody and counterpoint lines played when possible
4. Parallel fifths and octaves are forbidden
5. Avoid dissonant intervals - seconds, sevenths, and the augmented fourth

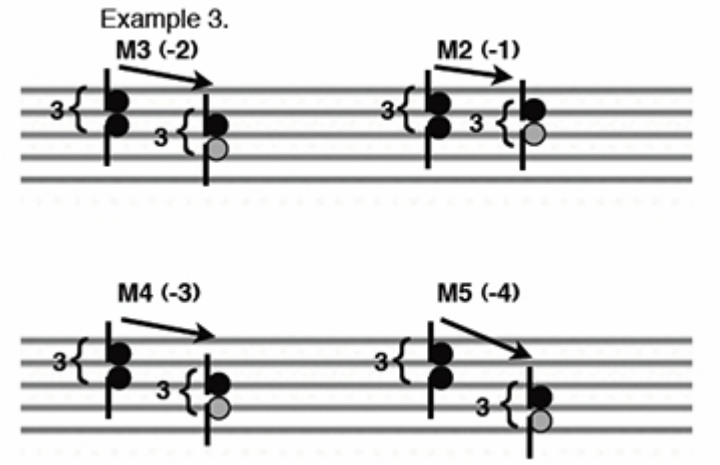


In example 1, the previous interval is 5. The new input melody note is a major second (M2, -1 whole note) value from the previous input note. In this case, the new output interval will be 3, corresponding to a major third and the gray note.

The states for the counterpoint FSM will be the previous interval. In example 1, that means a state of 5. The allowed intervals in first species counterpoint are 3, 4, 5, 6, 8, and 10.

The next note is chosen based on the state, and the interval between the new note in the melody and the previous note of the melody. There are 12 possible intervals: minor 2(m2), major 2(M2), minor 3(m3), major 3(M3), perfect 4(P4), augmented 4(A4), perfect 5(P5), minor 6(m6), major 6(M6), minor 7(m7), major 7(M7), and perfect 8(P8). Any interval greater than a perfect 8 can be reduced to one of these 12 intervals. The major counterpoint FSM will only properly deal with the major and perfect intervals. Example 2 illustrates what will happen in the case of a minor or augmented interval between the last melody input and the current melody input. For the sake of simplicity and memory, the minor and augmented interval will be mirrored by the counterpoint line, even if that breaks a law of counterpoint. Without this simplification, the number of allowed states would double, each state requiring its own case statement for every possible interval.

Given a major or perfect interval between the previous and current melody notes, the counterpoint fsm must choose a new interval following the rules of first species counterpoint. Luckily, not all of the cases must be enumerated separately. In example 3, the state is 3, if the new melody note is lower than the previous note, resulting in a negative melody interval, the new state will always be 3.



Every state and every interval will have to be defined. Without using any shortcut definitions like that of example three, there are 48 different instances for choosing the next state. This can be greatly expanded on by adding to minor chords to the allowed states and intervals.

Tone Generator (Chris):
This module is responsible for generating the signal that comprises the output sound waveform. It generates the counterpoint melody by taking an inverse Fourier transform of shifted frequency samples; the samples come from the FFT module, the frequency to shift to is provided by the counterpoint module, and the incoming dominant frequency is provided by the pitch finder module.

DSP/Filtering (Chris):

Equalizer:

This module is a 5-band equalizer that either attenuates certain bands of frequencies or boosts them relative to the other frequency bands in the audible spectrum. It can be implemented by running five different bandpass filters on five copies of the time-domain signal and then recombining them into one output.

Overdrive:

This module implements a type of distortion used to give a harsher perceived tone to the input audio, and is commonly seen in most popular music. It can be implemented, in a very basic sense, by amplifying the signal and then clipping the resulting waveform.

Reverb:

This effect can be accomplished by adding delayed, attenuated copies of the input signal into the output signal.

All of these filters will be tested within MATLAB and a modified version of the Lab 4 setup to ensure proper functionality. In addition, each filter can be switched on and off by the user according to individual taste; if an effect is turned off, the signal is instead sent through an all-pass filter instead of the corresponding effect filter.

Video Output (Chris):

Our visual component will be a control panel for our harmonizer. It reads an input from a PS/2 keyboard and reflects any changes the input makes on the various components of the system. For instance, the gain on each of the five bands of the equalizer will be represented graphically on the screen, each of which will be controlled by a pair of keys on the keyboard.

List of Parts:

- Speakers
- PS/2 Keyboard