

# ENGIN 100: Music Signal Processing

## PROJECT #3

### Music Synthesizer and Transcriber

Professor Andrew E. Yagle

Dept. of EECS, The University of Michigan, Ann Arbor, MI 48109-2122

#### I. ABSTRACT

This is the main project for the course. It has two parts: (1) Programming a simple music synthesizer entirely in Matlab using a Matlab musical GUI similar to the one you developed in Project #1, using snippets of real instruments; (2) Programming and evaluating a simple music transcriber entirely in Matlab using the Matlab stem-based staff-like notation in Project #1, but now including note duration information.

#### II. BACKGROUND

Music synthesizers can be as simple as recording the sounds of actual musical instruments and playing them back at variable speeds to obtain the desired pitches (wavetable synthesis), filtering of simple signals like square waves or sawtooth waves to achieve a desired effect (subtractive synthesis), or as complicated as actually computing each harmonic of a note separately and adding them together (additive synthesis).

Musical transcription, which generates sheet music (or the equivalent) directly from a musical recording, is much more difficult. It is by no means a completely solved problem, and present techniques involve signal processing concepts such as hidden Markov models that are graduate-level material, since real music is more complicated than the periodic signals whose periods change abruptly that we have considered in this course.

Nonetheless, transcription of simple music can be accomplished using the techniques presented in this course. You saw in Lab #3 that the spectrogram could itself function as a type of musical notation. Other approaches should be tried as well, such as the fft analysis at specific frequencies you used in Project #2.

The results of this project will be two .m files. One implements a music synthesizer which accepts either on-screen keys clicked with a mouse or a sequence of letters representing musical notes and outputs a melody using any of several musical instruments (guitars, clarinets, and others) selected from an on-screen menu. Of course, these can be added together (“mixed”) to achieve the effect of several instruments played together.

The other .m file implements a music transcriber which accepts a .wav file of music from your synthesizer and generates a musical-staff-like transcription using Matlab’s stem command. Details are provided below. Some suggested approaches for the transcriber will be discussed in lecture (see `proj3lecture.pdf`).

### III. PROJECT #3: WHAT YOU HAVE TO DO

Download the file `proj3.wav` from the course web site. This file contains snippets of length 32768 samples each from an electric guitar, a clarinet, a trumpet, and a single tone, all sampled at  $44100 \frac{\text{SAMPLE}}{\text{SECOND}}$ . Each instrument is playing the 13 notes of an octave (including the note at both ends twice) in succession.

Design your own instrument using additive synthesis (adding together harmonics with amplitudes selected by you) and label it with your team name. Also create a marching band sound by reverbing each trumpet note. Do this by adding copies of trumpet to itself, with each copy delayed slightly from the previous copy.

#### A. Synthesizer Specifications

Your synthesizer should have a keyboard, and note durations, like those shown below.

NOTE	1 second	WHOLE	HALF NOTES	QUARTER
LENGTH	44100	32668+100	16284+100	8092+100

  

- The final 100 samples of each note should be zeros, to provide duration information.
- The pull-down menu should offer a choice of instruments, prior to use of the keyboard.
- `H=icontrol('Style','Popup','Position',[500 250 100 50],'String','guitar|clarinet|trumpet|tone');`
- `pause;I=get(H,'Value');` This generates I=1,2,3,4 depending on the selection made.

#### B. Transcriber Specifications

- The transcriber should output a musical scale and notes like that of Project #1, except:
- The separation between transcribed notes should indicate the length of each note:
- A whole note followed by 3 spaces; a half note by 1 space; a quarter note by none.
- Use `reshape`, look for columns ending in zeros, let `T=[indices of those columns]`.
- Then you can use `stem(T,X)` where `X` is  $12 \cdot \log_2(\text{frequency}/440) + 6$  or whatever;
- The transcriber need not be able to handle the electric guitar (lower bass scale);
- Perform a **noise analysis** (error rate vs. SNR plot) as you did in Project #2.

#### C. Presentation

Your results will be presented in both written form (a technical report) and orally to the class and instructors, who will function here as your co-workers and bosses. You will be graded on the presentation as well as on the results. This is very realistic—the greatest invention since sliced bread is worthless unless you can explain to bosses and customers what it does and why it is valuable.