ENGIN 100: Music Signal Processing

PROJECT #2

Touch-Tone Synthesizer and Analyzer

Professor Andrew E. Yagle

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

I. ABSTRACT

Now that you have acquired some tools for analyzing frequencies and using Matlab, you will apply them to a small engineering project: Touch-tone phone tones. The goals of this lab are: (1) To analyze touch-tone phone signals and determine their spectral content; (2) to write a Matlab program that functions as a touch-tone keypad which generates the proper tones when pressed with the mouse (synthesizer); (3) to write a Matlab program that accepts as input a touch-tone phone signal, computes the phone number, and prints it out on the screen (transcriber); and (4) analyze the effect of noise on this transcriber. In addition to applying the tools you have acquired, this will also serve as a dry run for the final project to follow.

II. BACKGROUND

Touch-tone phones create a multi-frequency tone when a button is pressed. That tone is sent over a phone line (or wirelessly) as a signal. The goal of this project is to reverse-engineer the touch-tone system and build your own Matlab-based touch-tone synthesizer and transcriber from scratch. The only things you are allowed to use are: (1) the techniques you have learned so far in Engin 100; and (2) the 12 signals representing each button in the keypad of your touch-tone or cell phone, which are given in the file *proj2.wav*.

III. PROJECT #2: WHAT YOU HAVE TO DO

The results of this project will be two .m files, one implementing a touch-tone synthesizer, and one implementing a touch-tone transcriber. You also have to demonstrate to the lab IA that they work. You will do this by using the first program to write the signal to a file *touch.mat* (as well as making sound), and then using your transcriber to decode the signal stored in *touch.mat*. You will also have to use the spectral analysis techniques you have learned to analyze the touch-tone signals in the first place, just as you did with the musical tones, and study the effect of noise in the touch-tone signal on your transcriber.

A. Touch-Tone Signal Analysis

Use the techniques you have learned to analyze the 12 signals generated by the 12 keys on a touch-tone phone keypad. Download *proj2.wav* from the course website; this file contains, in succession, the signals produced by pressing keys "1,2,3,4,5,6,7,8,9,*,0,#" in that order for half a second each (total duration=6 seconds). All you will be told here is that you have the tools necessary to do this. Go to it!

B. Touch-Tone Synthesizer

Write a Matlab program (and store it as an .m file) that:

- Creates an on-screen keyboard using a sequence of `uicontrol` commands that resembles the 12-key keypad on a touch-tone phone or your cell phone (similar to what you did in Project #1);
- Produces the appropriate sound, lasting half a second, when pressed by clicking the mouse on it;
- Writes the signal to a file *touch.mat* for decoding by your transcriber.

C. Touch-Tone Transcriber

Write another Matlab program (and store it as an .m file) that:

- Accepts a touch-tone signal produced using the program above and stored in *touch.mat*;
- Prints out on the screen the phone number the signal represents (without the "-" in 123-4567);
- Need NOT be able to handle the "*" or "#" keys (these aren't part of a phone number).
- You *could* use `abs(fft())` and look for peaks in the spectrum of each digit signal; BUT:
- It is *much* faster to look *only* for those frequencies $\{F1 \ldots FM\}$ in which you are interested.
- Given: Row vector of sampled signal `X` where `N=length(X)` and F=sampling frequency, use:
- `C=X*cos(2*pi*[0:N-1]'*[F1...FM]/F);S=X*sin(2*pi*[0:N-1]'*[F1...FM]/F);Y=C.^2+S.^2`
- `[Z,I]=max(Y);` determines *location* I of maximum of Y. HINT: Use `rem(I+3*(J-1),11)`

D. Noise Analysis of Transcriber

Now analyze the effect of noise on your transcriber, as follows:

- Add noise to the signal produced by your touch-tone synthesizer, using `randn` (not *rand*);
- Compute the *Signal-to-Noise Ratio* SNR=$10\log_{10}\frac{\sum \text{signal}(n)^2}{\sum \text{noise}(n)^2}$. This is the noise level figure-of-merit;
- For each of 10 noise levels (multiply `5*randn` by successively larger numbers), estimate the *error rate* by counting the number of incorrectly-decoded digits out of 100. Plot the error rate as a percentage vs. SNR.

E. Project Report

Write up the results of your lab as a technical memo. Include:

- A diagram of the frequencies associated with each touch-tone key;
- The work you did in determining these frequencies; and the error rate vs. SNR plot;
- Printouts of the two .m files for your synthesizer and transcriber.

Also email your two .m files, named *yourteamname1.m* and *yourteamname2.m* to your lab IA.