EE430 - MATLAB TERM PROJECT

A MUSICAL INSTRUMENT TUNER

FINAL REPORT

I. INTRODUCTION

Music consists of sounds that attracts people's interests and makes them, somehow, feel good. Musical sounds, tones, have characteristics and the most important of these characteristics is the frequency of the tone. A tuner is a device that can detect the frequency of a short time sound and enables us to adjust our instruments to correct tones. This report includes the detail of a tuner implemented on MATLAB.

II. GETTING THE SOUND FROM ENVIRONMENT

First of all, the sound produced by the instrument must be recorded and then evaluation process comes. To record the sound wave, "wavrecord" command is used as follows:

Fs=44100;

tone=wavrecord(Fs\*4,Fs);

Here Fs is sampling frequency and Fs\*4 is number of samples in the sound wave recorded, i.e., the sound is recorded for 4 seconds. An example waveform is available in figure 2.1.

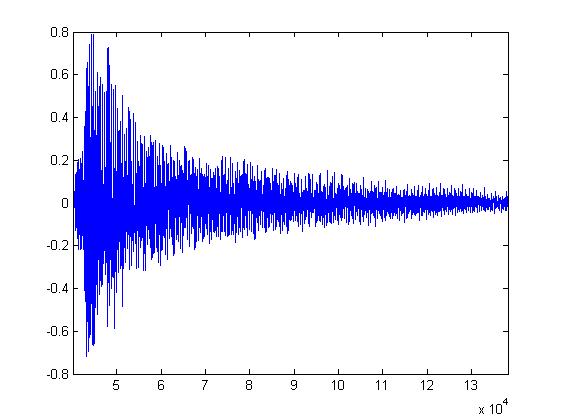


Figure 2.1.

III. EXTRACTING FREQUENCY INFORMATION

To get the frequency information, FFT is used with a proper windowing. Original signal is windowed by a Hamming window and then FFT is applied. This method yields the frequency content of the sound wave.

w = hamming(Fs\*4);

fftofsound=fft(tone.\*w);

After getting FFT, picking the index of maximum element in FFT will give us the fundamental frequency of the sound.

[~,peakfreq]=max(abs(fftofsound));

peakfreq=peakfreq/4;

An example of FFT and its peak is given in figure 3.1.

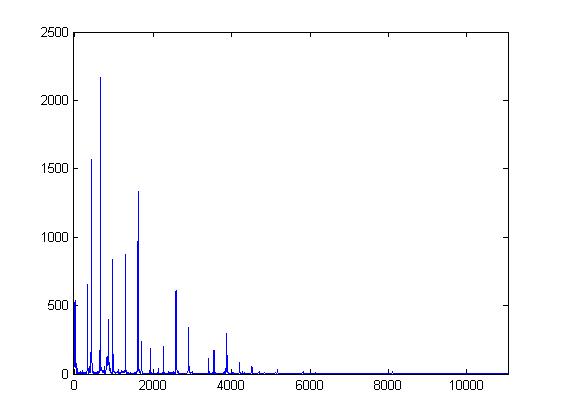


Figure 3.1.

IV. REPRESENTING THE RESULT

Finding the fundamental frequency by peak selection, we convert this frequency to letter notation by a proper algorithm and show the results in a GUI as seen in figure 4.1.

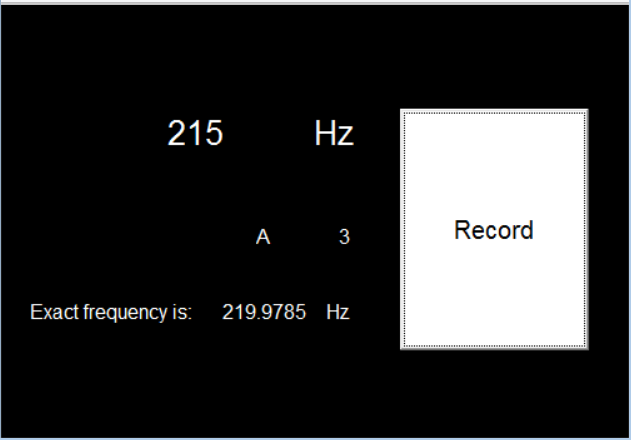


Figure 4.1.

In the above figure, when we press "Record" button, the sound is recorded for 4 seconds. Measured frequency, the closest letter to that frequency and its exact equivalence is shown on the screen so that we can tune our instrument correctly. Since we record for 4 seconds, 1/4=0.25 Hz is our frequency resolution, which is better than any human-being can be.

Whole MATLAB code can be found in appendix.

V. CONCLUSIONS & IMPROVEMENTS

We used peak selection for frequency estimation but this may bring some ambiguities. Due to the nature of the sound produced by a guitar, there are also harmonics along with the fundamental frequency and these harmonics might be higher than the fundamental one in amplitude. In such cases, peak selection may result in wrong frequencies and tuning may fail. Therefore, deciding the correct frequency by looking at its harmonics would be a better solution.

VI. APPENDIX - MATLAB CODE

function record\_Callback(hObject, eventdata, handles)

% hObject handle to record (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

global tone;

double k;

int i;

Fs=44100;

tone=wavrecord(Fs\*4,Fs);

w = hamming(Fs\*4);

fftofsound=fft(tone.\*w);

figure;

plot(abs(fftofsound));

[~,peakfreq]=max(abs(fftofsound));

peakfreq=peakfreq/4;

set(handles.result,'String',num2str(peakfreq));

set(handles.hz,'String','Hz');

k=round(12\*log2(peakfreq/16.35));

set(handles.actual,'String','Exact frequency is: ');

set(handles.actualfreq,'String',num2str(16.35\*(2^(k/12))));

set(handles.hz2,'String','Hz');

i=0;

while(k>=12)

k=k-12;

i=i+1;

end

set(handles.number,'String',num2str(i));

if(k==1||k==3||k==6||k==8||k==10)

set(handles.diyez,'String','#');

else

set(handles.diyez,'String','');

end

if(k<5)

k=floor(k/2);

else

k=ceil(k/2);

end

set(handles.nota,'String',char(65+mod(k+2,7)));

AYKUT DEMİREL - 1813948 - Section: 2