Decipher a Music into Musical Notes with Fast Fourier Transform using MATLAB

Project by: Emad Mohamed Mahdy

Intro

In this project, you are given a simple (piano only) version of a famous music clip: **Eine kleine Nachtmusik (Serenade No. 13 for strings in G major), K. 525**, which is a 1787 composition for a chamber ensemble by Wolfgang Amadeus Mozart.

See: https://en.wikipedia.org/wiki/Eine_kleine_Nachtmusik.

Your task is to produce a MATLAB script that can decipher the notes of this music clip on simple version. The output should be an excel file (at the end) that stores all the notes and their time slots. The detection of notes should be in real-time (while the music is played) shown on screen

watch: https://www.youtube.com/watch?v=sbLsd98eLJs

Some useful hints:

- The music files are originally sampled in CD quality (44.1 KHz). For analysis, you don't need this resolution, you can "decimate" to 16KHz or even up to 8KHz.
- Each note is a "quick event" in time. So, to capture a single note, you need to window the signal, possibly with "overlapping" windows with a reasonable duration.
- You should use only have FFT (Fast Fourier Transform).
- For note prediction, this can be useful: https://pages.mtu.edu/~suits/notefreqs.html
- When you accomplish the analysis and predict which note it is played at a given time interval [t, t+dt], put it on the screen. Here you need to perform some nice GUI so that your client

can easily see this preferably over the waveform or spectrogram of the signal (and listening the music alongside

Project implementation steps using MATLAB

- First read input music and store it in variable In_File.

```
In_File='SIMPLE eine kleine nachtmusik - mozart.wav';
```

- sample the music input and store sample in vector y
  with sample rate Fs (41100 sample pre sec)

```
[y , Fs] = audioread (In_File);
```

- import all music tone symbols from excel sheet and store it in a table T.

```
T = readtable('ScaleFreqs.xlsx','Range','A4:B112');
```

- divide all samples to groups each group will be .5 second length using for loop.

```
End=length(y);
Sample_Period=Fs/4;
i=1;
 for v = 1:Sample_Period:End-1
```

- store each group of samples in Sound_Signal variable and display sound each iteration

```
Sound_Signal=y(v:v+Sample_Period,1);
sound(Sound_Signal,Fs);
```

- extracting the current sound part of in this sample period doing Fast Fourier Transform (FFT) of input signal to convert it to frequency domain

```
Input_Signal=y( v+1500 :  v+Sample_Period-1500 , 1 );
N=length(Input_Signal);
N1=ceil(N/2);
```

```
INPUT_SIGNAL=fft(Input_Signal);

INPUT_SIGNAL_abs=abs(INPUT_SIGNAL);

INPUT_SIGNAL_abs_double=fftshift(INPUT_SIGNAL_abs);

freq_axis=[0:N-1];

freq_axis_double=(freq_axis-N1)*Fs/N;
```

- after transforming signal to frequency domain, we will search for the frequency of the musical tone in this part of samples

first, we search for the maximum amplitude of this sampled part

```
[Max,Index]= max(INPUT_SIGNAL_abs_double);
```

then, we use this index to find the frequency of this maximum amplitude

```
FREQ=freq_axis_double(Index);
```

then we search for this frequency in the table of musical notes and cross-pending frequencies.

```
rows = (T.Var2<=abs(FREQ(1,1))+3&T.Var2>(abs(FREQ(1,1))-3));
```

extract the frequency and its cross-ponding piano tone to variable x

```
x=T(rows,:);

temp=x.Var1;
```

- calculate the current time slot of current iteration

```
t=v/Fs;

t=round(t,2);

Time(i,1) = num2cell(t);
```

- display the time slot and corresponding musical tone in MATLAB GUI

```
set( handles.Time,'string' , t );

If Max>=10

    set(handles.Note,'string',temp);

else

  set(handles.Note,'string','Selience');
```

- print time slot and musical tone extracted from audio files to excel sheet

```
xlswrite('Out_File.xlsx', Time);
xlswrite('Out_File.xlsx',Note,'B1:B224');
```

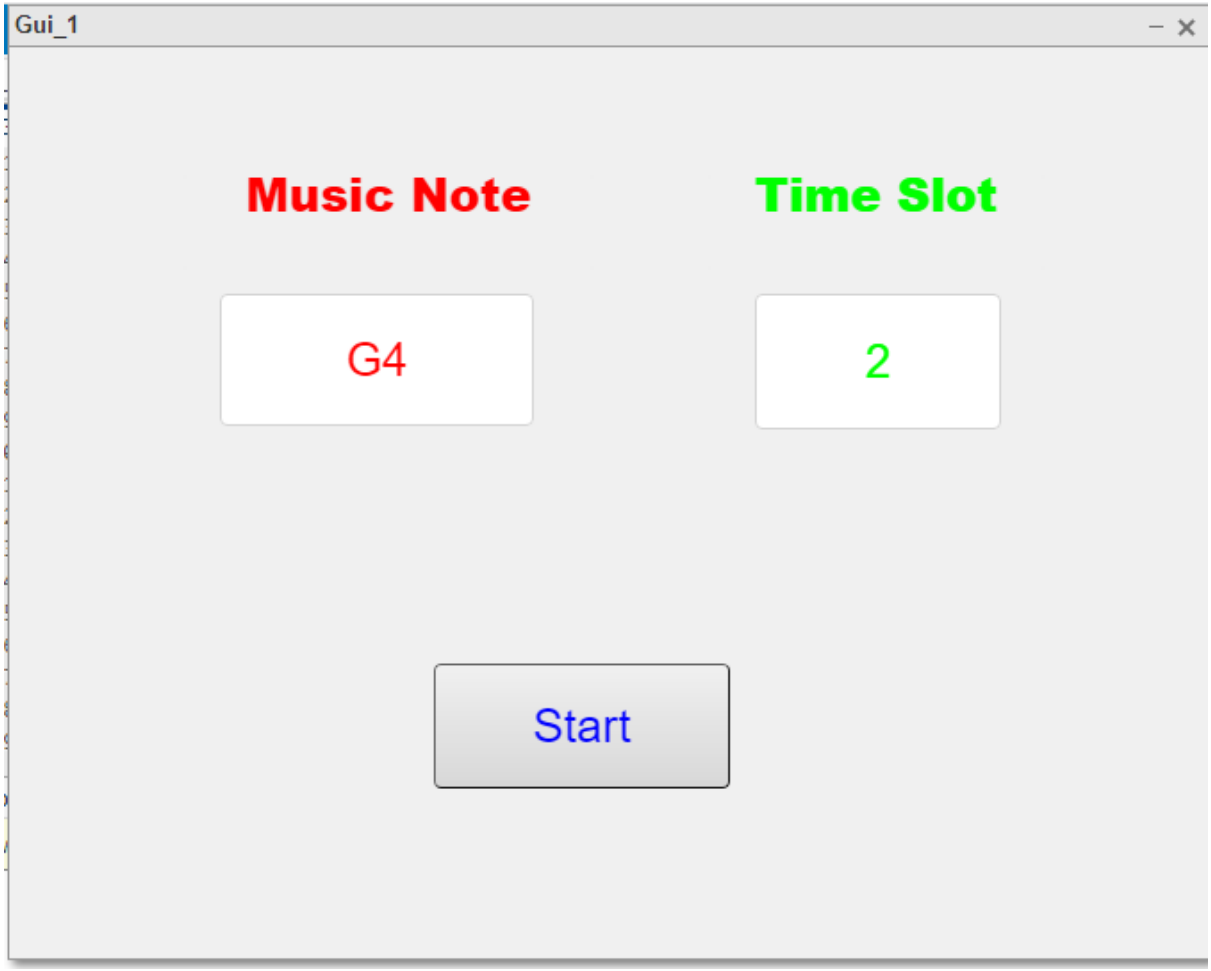

Figure 1 snap shot of gui of this project