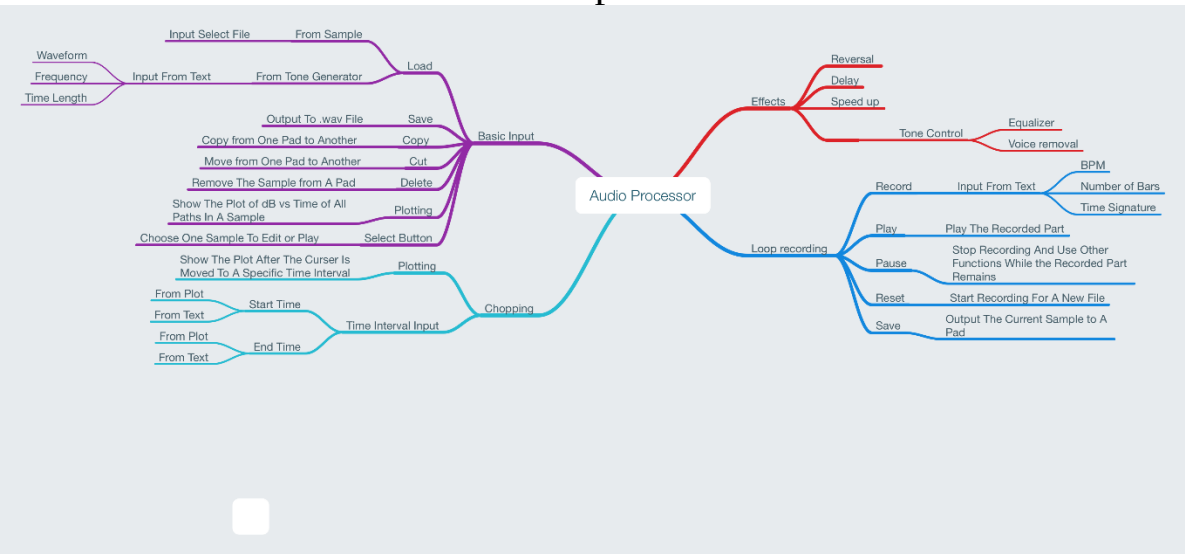
Audio Project Report

α-UMi

Purpose:

In this project, we developed a program that is used to process audio files with various functions using matlab. Here is a graph containing all the functionalities that we have developed for now.

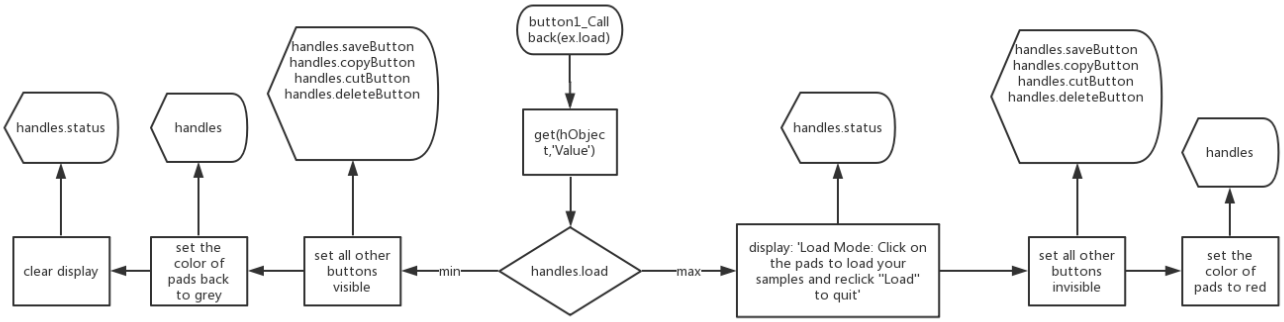


Procedure:

Here are the flow charts of the most representative functions contained in our projects. Each chart may apply to more than one functions in our project.

1) The Basic Input Buttons:
   - Load
   - Save
   - Copy
   - Cut
   - Delete

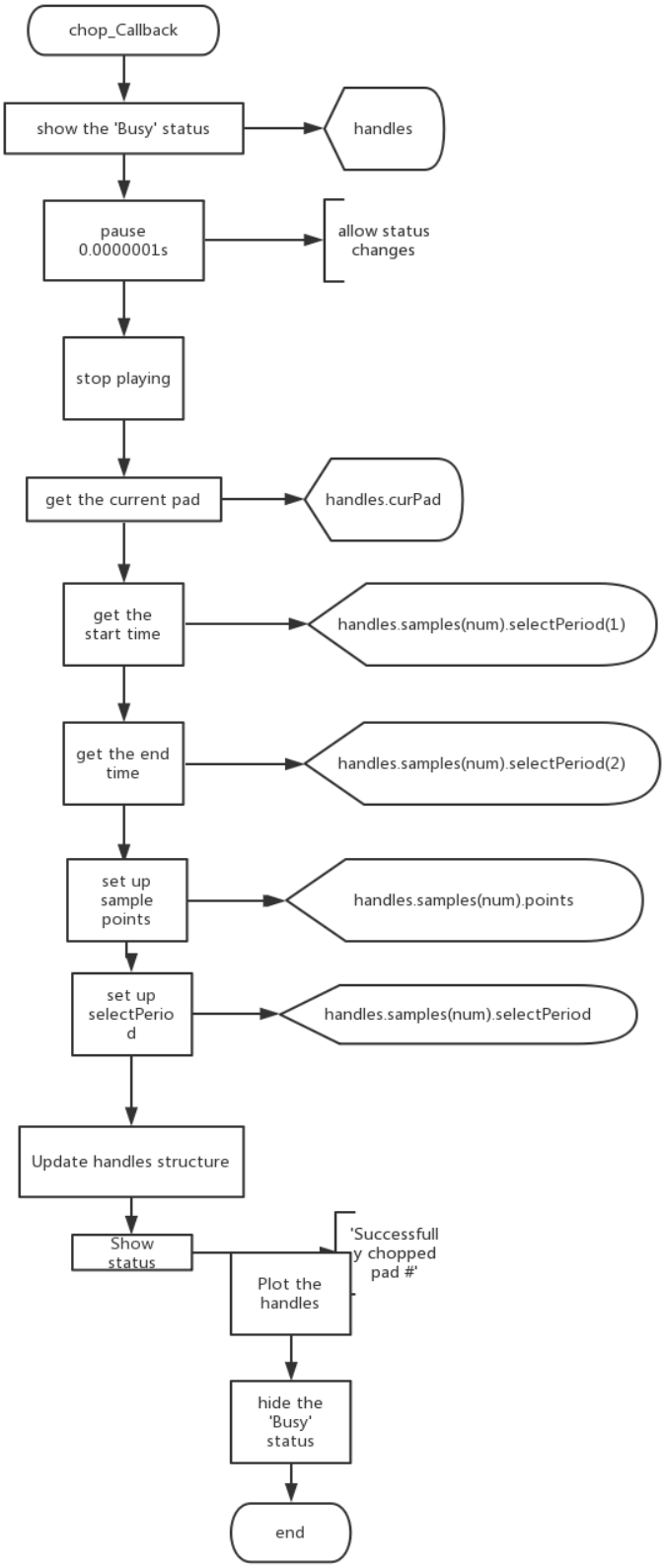
This chart is using load as an example:



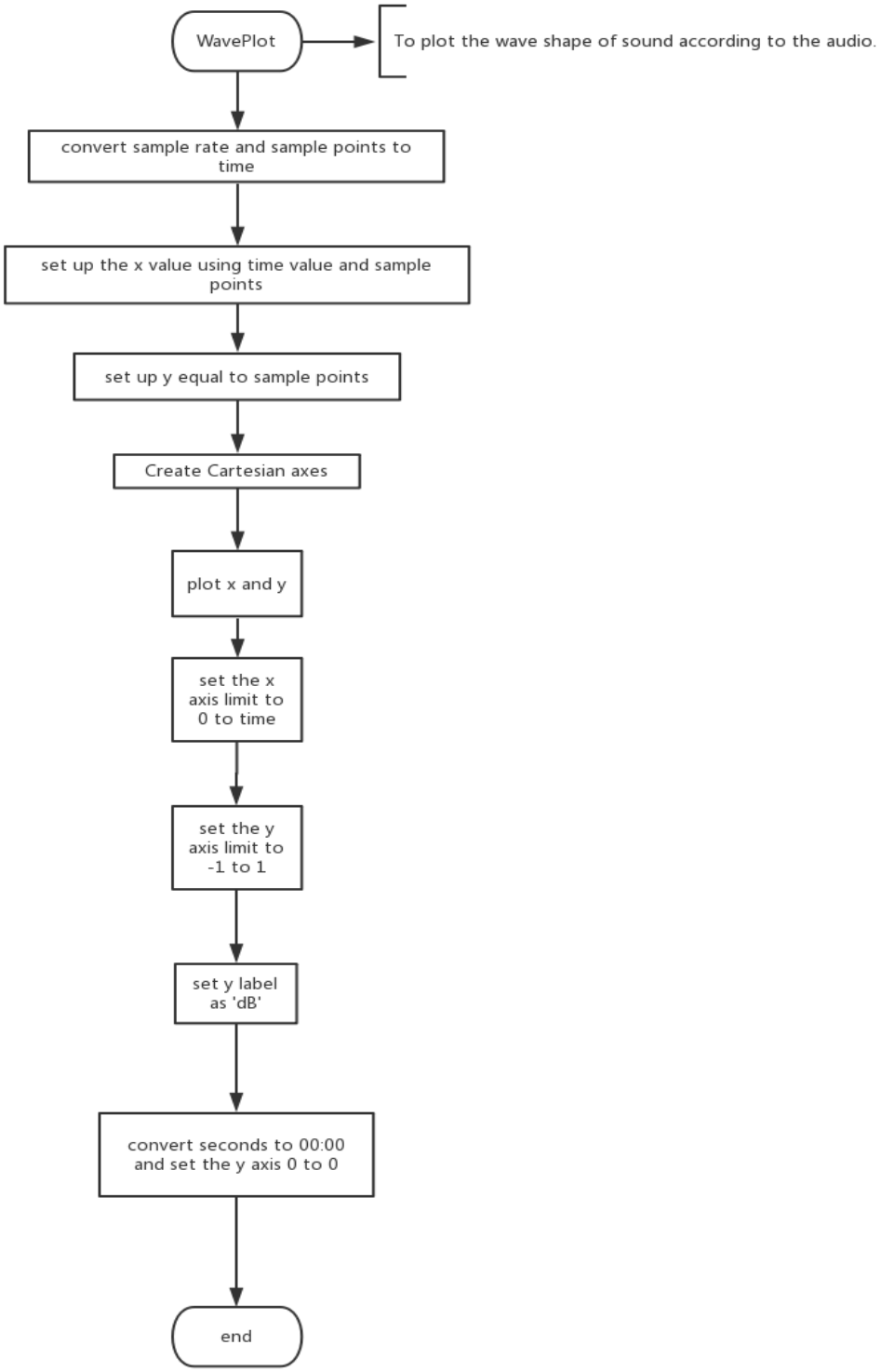
2) The Pad Callback:
   Pad 1 to 16


pad_Callback
handles.loadButton
max
LoadToPad(hObject,handles,padnumber)
min
handles.copyButton
max
CopyFromPadToPad(hObject,handles,padnumber)
min
handles.cutButton
max
CutFromPadToPad(hObject,handles,padnumber
min
handles.button_padnumber
max
EmptyPadStatus(handles,padnumber)
min
handles.saveButton
max
SaveFromPad(handles,padnumber)
min
handles.deleteButton
max
DelFromPad(hObject,handles,padnumber)
min
SelectButton(hObject, handles, padnumber)
PlayFromPad(handles, padnumber);
end

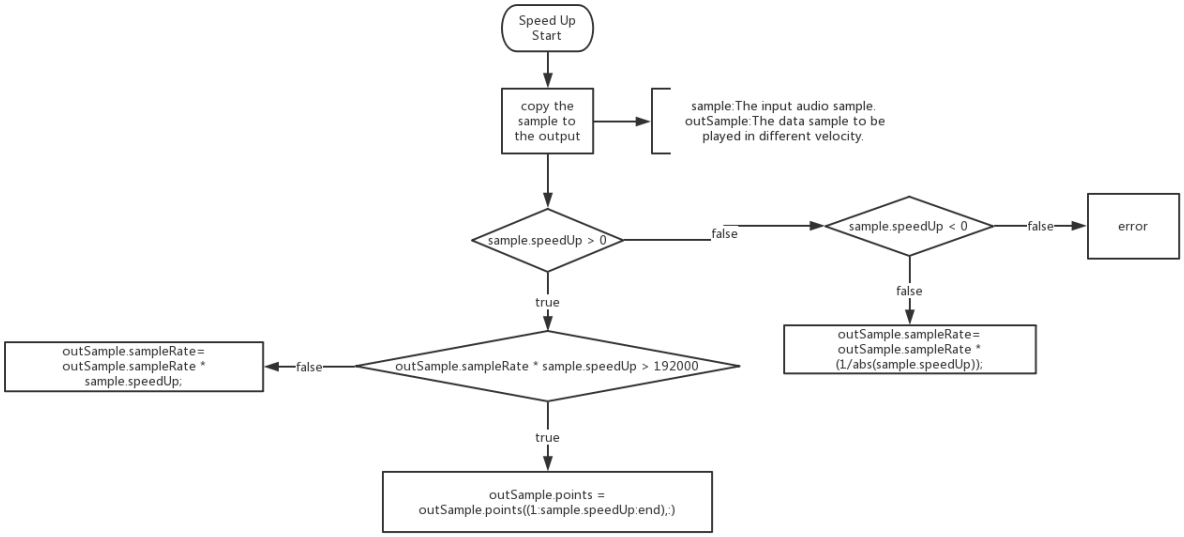
3) The Chopping Function Callback



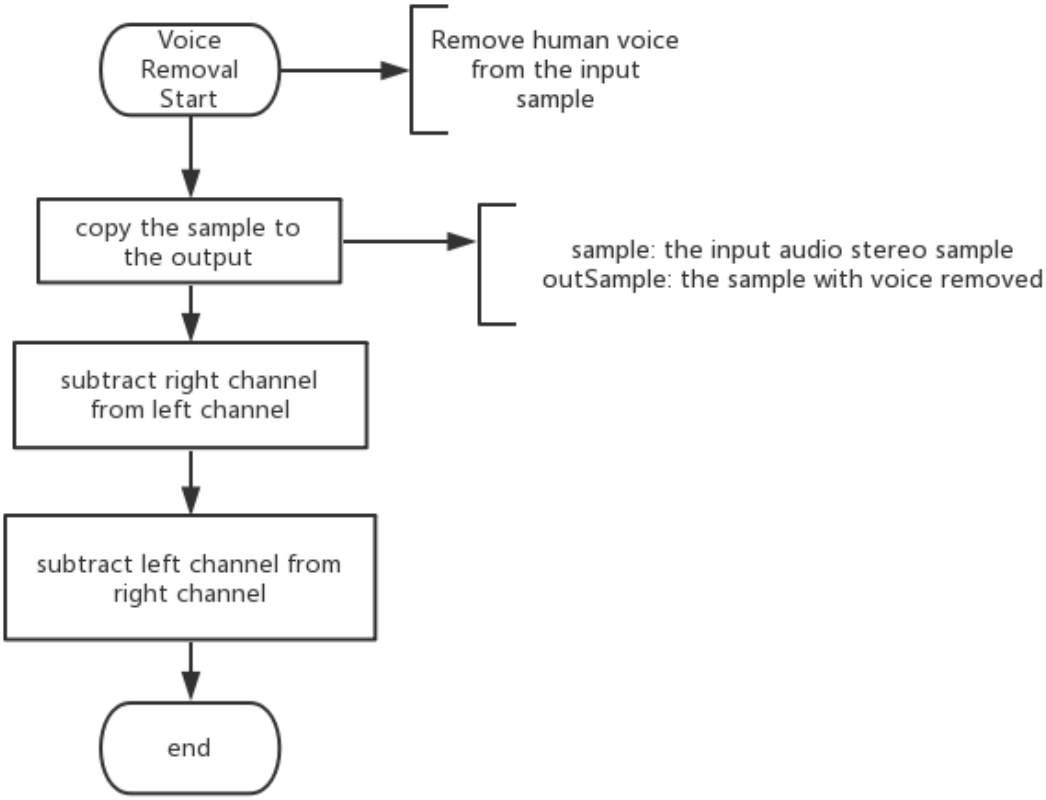
4) The Wave Plotting Function



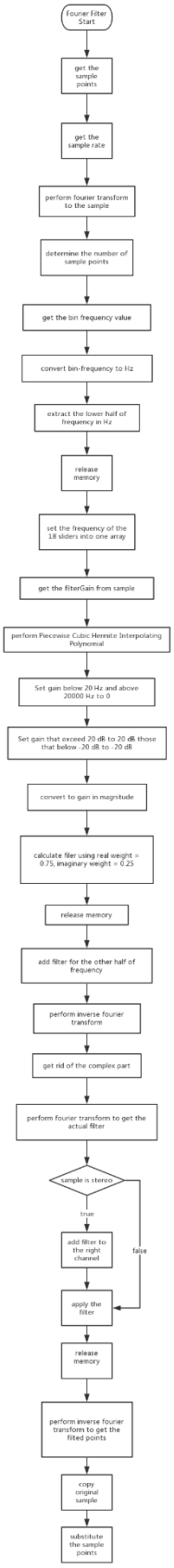
5) The Speed Up Function



6) The Voice Removal Function

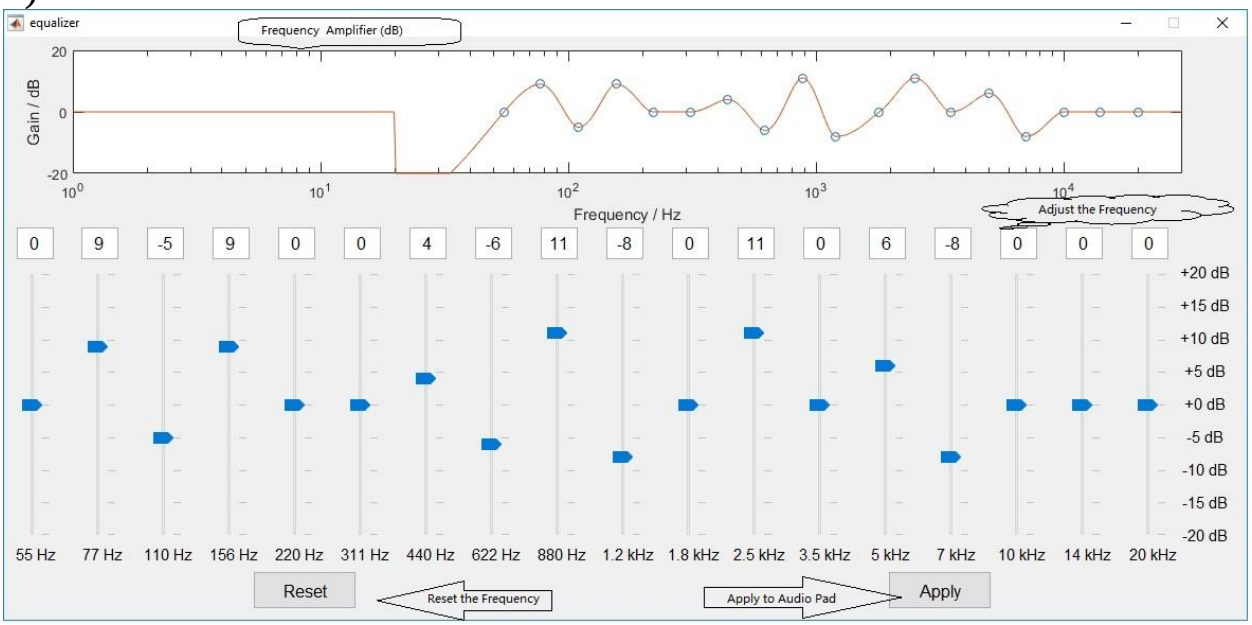


7) The Fourier Transformation

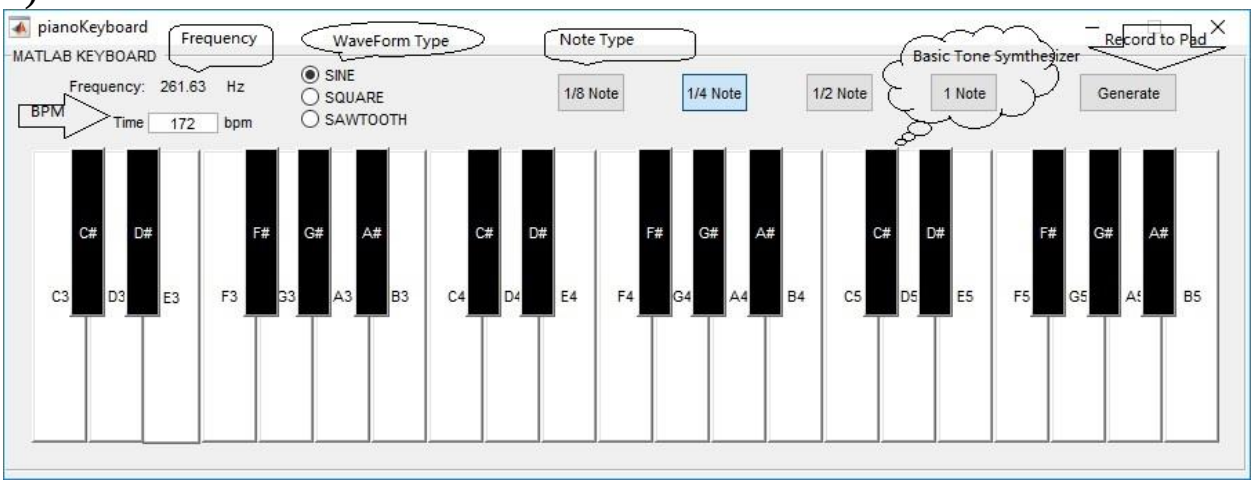

Fourier Filter Start
get the sample points
get the sample rate
perform fourier transform to the sample
determine the number of sample points
get the bin frequency value
convert bin-frequency to Hz
extract the lower half of frequency in Hz
release memory
set the frequency of the 18 sliders into one array
get the filterGain from sample
perform Piecewise Cubic Hermite Interpolating Polynomial
Set gain below 20 Hz and above 20000 Hz to 0
Set gain that exceed 20 dB to 20 dB those that below -20 dB to -20 dB
convert to gain in magnitude
calculate filer using real weight = 0.75, imaginery weight = 0.25
release memory
add filter for the other half of frequency
perform inverse fourier transform
get rid of the complex part
perform fourier transform to get the actual filter
sample is stereo
true
false
add filter to the right channel
apply the filter
release memory
perform inverse fourier transform to get the filted points
copy original sample
substitute the sample points

The GUIs that we have designed:

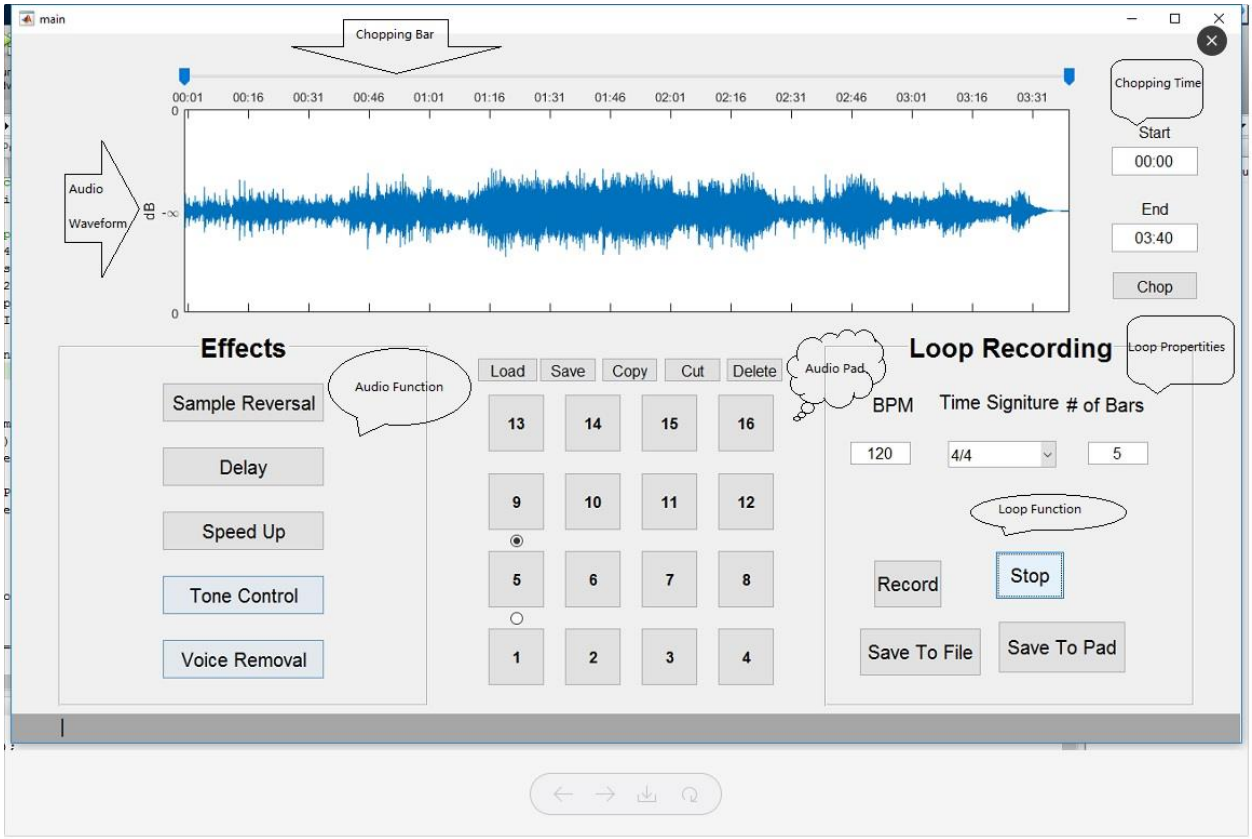
1)



2)



3)


main
Chopping Bar
00:01
00:16
00:31
00:46
01:01
01:16
01:31
01:46
02:01
02:16
02:31
02:46
03:01
03:16
03:31
Chopping Time
Start
00:00
End
03:40
Chop
Audio
Waveform
dB
Effects
Audio Function
Sample Reversal
Delay
Speed Up
Tone Control
Voice Removal
Load
Save
Copy
Cut
Delete
Audio Pad
13
14
15
16
9
10
11
12
5
6
7
8
1
2
3
4
Loop Recording
Loop Propertities
BPM
Time Signiture
# of Bars
120
4/4
5
Loop Function
Record
Stop
Save To File
Save To Pad

Appendix

Here are the names and initial values of the handles existing in our project:

```
handles.samples(i).points = [];            % the sample points
handles.samples(i).sampleRate = 0;    % the sampling rate
handles.samples(i).selectPeriod = [];    % used for plotting and chopping
handles.samples(i).filterGain = [];        % data for Fourier Transform
handles.samples(i).isReversed = false;    % Boolean for sample reversal
handles.samples(i).delay = 0;                  % data for delay
handles.samples(i).speedUp = 0;               % data for speedup
handles.samples(i).isVoiceRemoved = false;      % Boolean for voice remove
handles.samples(i).origSample = handles.samples(i);    % copy of original sample
handles.curPad = 0;        % current pad
handles.toneSample = [];
handles.slider = jRangeSlider;
```