ECE-GY 6183 : DSP Lab
Project BEATRIX

Shubham Shandilya
New York University
ss15590@nyu.edu

Tanisha Madhusudhan
New York University
tm3805@nyu.edu

Contents

1 Introduction **1**

2 Library requirements **1**

3 The Beatrix features **2**

3.1 The Drum pads: 2
3.2 The Play/Pause button: 2
3.3 The Tempo button: 2
3.4 The Beats in loops button: 3
3.5 Save Menu 3
3.6 Load Custom beats: 4
3.7 Room Impulse: 4
3.8 Speaker Layout 5
3.9 Fourier spectrum audio-visualizer : 6

1 Introduction

Using the powers of the Python libraries: `PyGame, PyAudio and SciPy`, **BEATRIX** an intuitive rhythmic drum beat sequencer loops over to generate drum patterns and experiment with it. It provides an interactive GUI to allow the users a real-time tool to play and configure various parameters to generate the desired melody.

2 Library requirements

PyGame is a free and open-source cross-platform library for the development of multimedia applications like video games using Python. It uses the Simple DirectMedia Layer library and several other popular libraries to abstract the most common functions, making writing these programs a more intuitive task.

Following are the libraries required to execute the BEATRIX: PyGame, Pyaudio, Scipy, numpy, matplotlib, wave and soundfile.

3 The Beatrix features

3.1 The Drum pads:

A grid of pads is laid out against each instrument as shown in the figure 1 given below to be selected and configure to create a melodious sequence. On clicking the colour the pad changes from default `LIGHT_GRAY` to one of the ever-changing `VIBGYOR` shades.

Figure 1: Drum beat Grid pads

In case, the entire instrument needs to be de-selected, the instrument pad on the left-size could be selected to toggle in order to activate/de-activate. The colour of the previously active pads would be changed to `DARK_GRAY`.

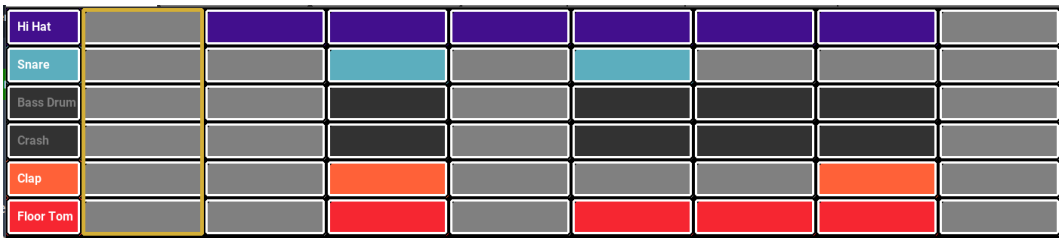

Figure 2: Deactivated Bass Drum and Crash instruments

3.2 The Play/Pause button:

Once the beats are configured, the `Play/Pause` button could be tapped on to make the configured beat play the music out loud on the speakers.

Figure 3: Play/Pause button

3.3 The Tempo button:

In situations where the tempo of the playback is not satisfactorily, the user can either choose between the given presets or increase/decrease in order of five. The current BPM could seen as displayed in `RED`.

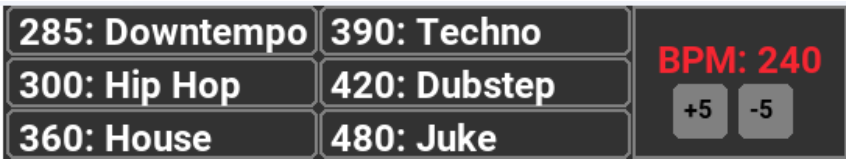

Figure 4: Tempo presets and BPM controls and display

3.4 The Beats in loops button:

Manage the beats controls in order to increase/decrease the beats in the given loop.

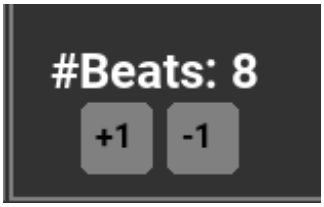

Figure 5: Number of beats in one given loop

3.5 Save Menu

Ultimately, when the sound thus creatively constructed is melodious enough or have to be saved and worked upon in a later time, the BEATRIX offers a feature to save the current pad configuration, room setting, BPM and Beats in the loop value, speaker layout and even activated/de-activated instruments and other parameters with a custom name when clicked upon 'SAVE' button. All the mentioned parameters are saved in a text file and can be easily loaded back through 'Your Custom beat' button when required.

Figure 6: 'Save' Menu

3.6 Load Custom beats:

When in need to load back the beat configuration and other parameter which the user might be working on previously, it can be done by clicking on the *Your Custom beat*. Upon clicking the same, the Load Menu will pop up and the previously saved beat with their given names could be loaded.

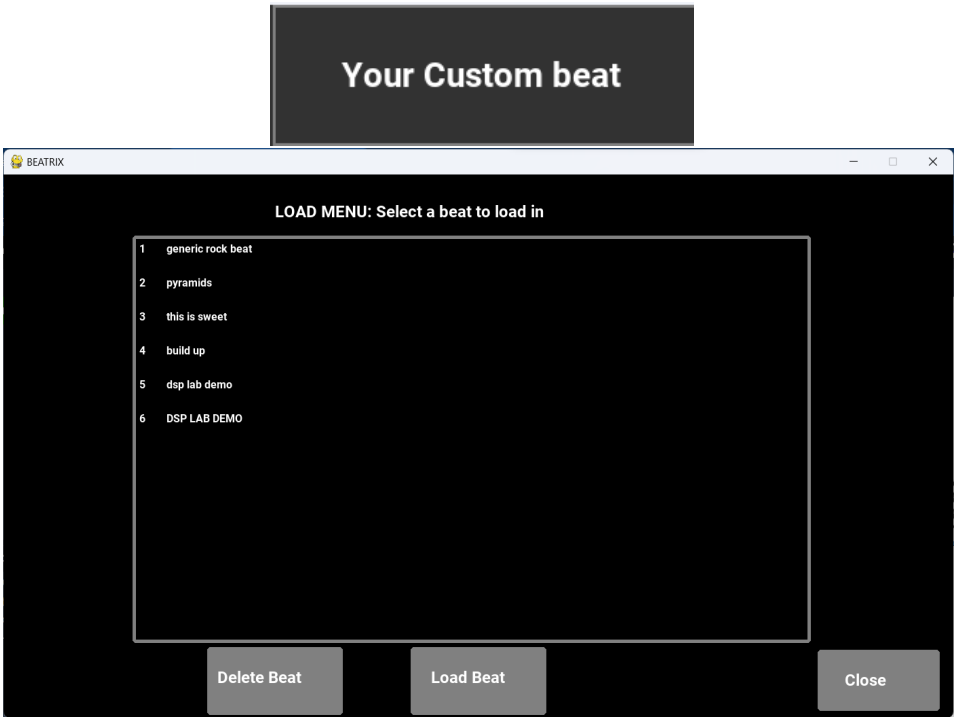

Figure 7: Load Custom Beats

3.7 Room Impulse:

The room impulse response is the *transfer function* between the sound source and microphone. In order to recover the original sound source, the received microphone signal can be convolved with the inverse of the room impulse response function. Generally, the system can only be approximated because it is rarely minimum-phase, i.e. causal and invertible.

There are several approaches to obtaining an estimate of the room impulse response transfer function. One approach is to use cepstrum analysis [1]. Cepstrum is the Fourier transform of the log spectrum, DFT(log(X(ω))). This is a measure of the frequency of variation in the log spectrum. Speech is considered slowly varying relative to the reverberant components in the log spectrum. Therefore, the speech and transfer function components can be separated.

Another approach to estimating the room impulse response transfer function is to use the linear prediction (LP) residuals [2]. Clean speech components cause the LP residuals to remain close to zero, while reverberation causes the the residuals to be time-varying. Thus, reverberation lowers the kurtosis of the probability distribution of the LP residuals relative to clean speech. For dereverberation, the objective function of the adaptive filter to remove the reverberent speech components is to maximize the kurtosis of the linear prediction residuals.

Reverberation is an audible effect that is heard every day as a result of sound interacting with the environment around the listener. It is an effect that is commonly added artificially to electronically recorded and produced music, as well as in other media such as cinema, virtual reality applications and computer games. In order to achieve the said reverberation using the impulse function of twelve such interesting buildings, spaces and other sources, these functions were downloaded from OpenAir website and then convolved with the each instrument selected to be played to get the desired effect.

Figure 8: Twelve Room Impulses

3.8 Speaker Layout

The audio stream is generated as a `mono-channel` by default. As and when the need arises to create a spatial audio effect:

- the Left speaker could be clicked to play the music through just left speakers.
- the Center speaker could be clicked to play the music through both the speakers.
- the Right speaker could be clicked to play the music through just right speakers.

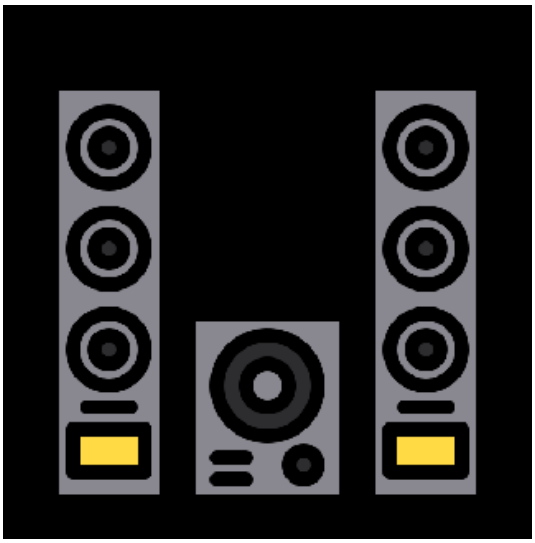

Figure 9: Speaker layout

3.9 Fourier spectrum audio-visualizer :

In order to visualize the music thus being played by the Beatrix, the audio from the speakers shall be sampled by choosing the device which has the configuration of *Stereo Mix* in it.

The `PyAudio Stream`s uses the **callback mode** to generate audio data on the fly or immediately process recorded audio data. PyAudio calls the specified callback function whenever it needs new audio data available. The `callback function` is designed to correct the format of the audio stream into `np.int16` from the buffer and estimates the frequency and power spectral arrays through periodogram using scipy's `sig.periodogram` module. Note that PyAudio calls the callback function in a separate thread.

We started processing the audio stream using `pyaudio.Stream.start_stream()`, which will call the callback function repeatedly until that function returns `pyaudio.paComplete`.

Sampling with the `Block_size = 256` and sampling rate of `44.1 KHZ`, square-root of the average 'volume' or power for each frequency range is used to compute the amplitude of the waveform with a notion of instant growth and proportional decay. This is plotted against the frequency spectrum. The plot is eventually re-drawn at every iteration and converted to `RGBA buffer` and placed on the surface to have the real-time visualization.

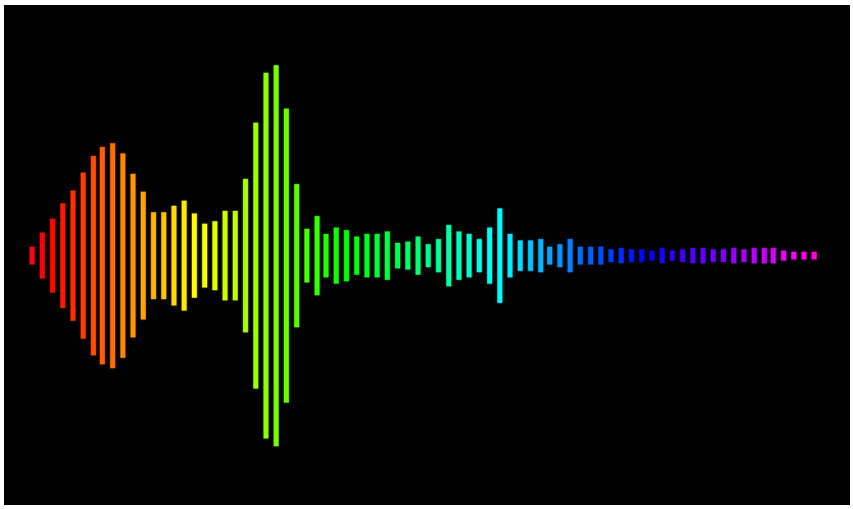

Figure 10: Twelve Room Impulses

References

[1] Childers, D.G., Skinner, D.P., Kemerait, R.C.: The cepstrum: A guide to processing. Proceedings of the IEEE **65**(10), 1428–1443 (1977)

[2] Prasanna, S.M., Gupta, C.S., Yegnanarayana, B.: Extraction of speaker-specific excitation information from linear prediction residual of speech. Speech Communication **48**(10), 1243–1261 (2006)