Minji Kim (minjik4)

15-112

Term Project Design Proposal

**Project Description**

Title: Python Audio Visualizer

Description: The objective of this project is to create an audio visualizer using pygame, pyaudio, and aubio. The product will be able to import a media file, analyze the music using Fourier Transform, and display an animation that corresponds to the music. It will also be able to perform the same task based on real-time audio input, where the user can use a microphone to input sound and generate an animation.

**Competitive Analysis**

A similar project I have seen online was a past term project called “Impulse”. This project uses live audio visualization to shoot out circles from the center of the screen, which appear in different colors according to the pitch of the sound. It implements a multiplayer game, where the user is able to tap the keyboard according to the appearing beat circles. It also generates animation in real-time using microphone audio input. Lastly, the project generates a random audio visualization by importing a randomly selected music from Soundcloud.

Another similar project is a past term project called “Burst”. This project is also an audio visualizer that is built on a processing environment. A cube in the center of the screen represents the intensity of the music, and frequency domain obtained through Fourier Transform analysis is displayed on the four corners of the screen. It also has the feature of visualizing audio in real-time, accepting user sound inputs through the microphone.

There are five main dimensions for this type of project (in order of significance):

1. Beat Detection

: a user is able to see a visualization of beats in a music. An audio visualizer is able to detect beat, as the visualization’s core feature is to demonstrate a rhythm of an audio by visualizing the beats.

2. Pitch Detection

: it is also crucial for an audio visualizer to be able to animate differences among pitches. This can be done through variation of colors, intensity of the animation, speed of an animated object, etc.

3. Music Library / Selection

: a user is able to change and select songs to visualize with the project. The project provides its own library of music to select from. It is significant that the project adapts to different kinds of music that the user selects.

4. Live Feature

: the program is able to generate real-time animation using the user input from the microphone. The animation uses the same algorithm to generate animation.

5. Random Music Import

: the program is able to import random music to visualize from the web, using a popular music platform.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Beat Detection | Pitch Detection | Music Library / Selection | Live Feature | Random Music |
| Impulse | Yes | Yes | Yes | Yes | Yes |
| Burst | Yes | Yes | Yes | Yes | No |

**Structural Plan**

* Display (main file)
  + Data
    - detect\_beat() function
    - detect\_pitch() function
    - fft() function that will do the fourier transform analysis
    - analyze\_beat() function that takes in data from detect\_beat()
    - analyze\_pitch() function that takes in data from detect\_pitch()
    - functions that transform the data from functions above into graphical data
      * color
      * shape
      * size
      * speed
  + Animation
    - draw\_center()
    - draw\_rings()
    - draw\_particles()
    - draw\_freqGraph() that takes in values from fft() and draws a graph on frequency domain for visualization
  + Threading
    - Thread the Animation and Music part together to play simultaneously
* Music (directory file)
  + Set of music stored in .wav format

**Algorithmic Plan**

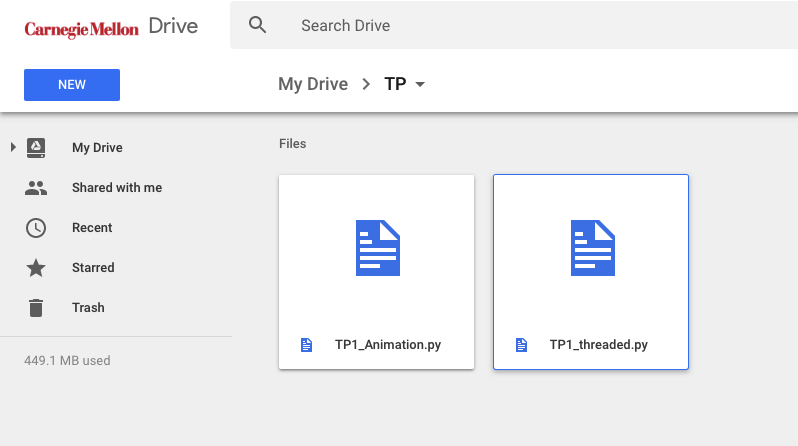
1. Generate an analysis of current beat and pitch, which can then be put into different categories (buckets)
   1. Use aubio to detect the time of the beats
   2. Use aubio to determine the pitch
   3. Generate a dictionary (or list) of beats and pitches
2. Transform the data of sound into data for graphics, which can be visualized
   1. Color
   2. Shape
   3. Size
   4. Speed of animated objects
3. Using the graphical data derived from the sound analysis, create objects on display for audio visualization
   1. Use pygame to draw animated objects
   2. Using the dictionary (or list) of beats and pitches, determine the quantity of each object

**Timeline Plan**

* TP1: Basic beat and pitch detection, simple animation based on beat and pitch
* TP2: Higher level animation, Fourier Transform Analysis, real-time analysis using pyauido and aubio
* TP3 (Final): music library, implement Fourier Transform into animation and finalize the rest of the animation

**Version Control**

I will implement version control in this project by using google drive. I will store all my codes in the google drive and save any modifications made in order to be able to access past versions easily.



**Modules**

This project will be using the following modules:

* pygame
* pyaudio
* aubio
* threading

**TP2 Update**

Instead of beat detection, I have decided to add the function for intensity detection, which is more relevant for audio analysis. This will be done using the pyaudio module, while pitch detection remains the same. Moreover, the main focus of my project will be on live audio visualization, and importing music and visualizing the imported audio file will be a supplemental function for my project.

**TP3 Update**

I completed the function for Fourier Transform Analysis and also wrote a function that draws a live spectrogram of the frequency domain on the screen according to the audio input.

I also added a beat detection function, written according to a popular beat detection algorithm using local and instant energy analysis. This way, the visualizer’s reaction to beat is much more accurate.

I have implemented loading a music and playing it in the background while running the visualization on pygame. A user can select music from the given list of sample songs in the program.

I have improved user interface by adding two more screens: start screen and select screen. Start screen includes the live spectrogram and buttons that lead to two different modes of the project. The select screen includes three buttons that each lead to a sample song visualizer. All animation screens now have a back button that leads to the start screen.

I have significantly updated the graphics component for the audio visualizer. Here are the things I added or modified to improve the graphics:

* Add particle objects with varying colors. Particle objects are images drawn and created on my own.
* Add spectrogram onto the screen.
* Add core object onto the screen that expand and shrink according to the intensity.