Lab Report: Random Audio Playlist Player using MoviePy

Aditya Varun V

May 18, 2023

1 Abstract

The aim of this lab report is to provide an analysis of a Python script that implements a random audio playlist player using the MoviePy library. The code allows users to create and play a playlist of audio files in a random order. This report provides an overview of the code structure, explains the key components, and discusses its functionality.

2 Introduction

The provided code is a Python script that utilizes the MoviePy library to create a random audio playlist. The script allows users to specify the number of times they want the playlist to be played and randomly selects songs from a predefined list. The audio files are extracted from MP4 files using the VideoFileClip class provided bgy the MoviePy library.

3 Methods

The code can be divided into several sections:

3.1 Importing Required Libraries

The script begins by importing the necessary libraries, including `numpy` and `moviepy.editor`. These libraries provide functions for numerical operations and working with video and audio files, respectively.

3.2 Defining the `play_mp4_audio()` Function

The `play_mp4_audio()` function is responsible for playing the audio extracted from an MP4 file. It takes a filename as input, creates a VideoFileClip object from the file, extracts the audio using the `audio` attribute, and plays the audio using the `preview()` method.

3.3 Initializing Variables

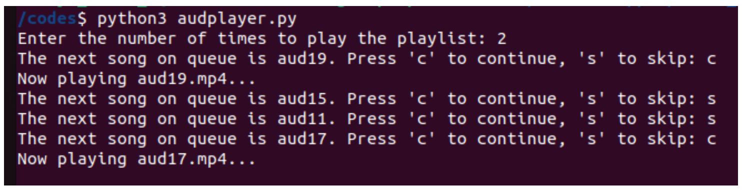
Several variables are initialized before the main execution loop:

- `played`: A numpy array of size 20, initialized with zeros to keep track of which songs have been played.
- `list_of_songs`: A numpy array containing numbers from 1 to 20, representing the song playlist.
- `c`: A counter variable to keep track of the number of songs played in the current round.
- `r`: A counter variable to keep track of the number of times the entire playlist has been played.
- `x`: User input to specify the number of times the playlist should be played.

3.4 Playing the Playlist

The main execution loop is implemented using a `while` loop. The loop continues until the number of rounds (`r`) reaches the specified input value (`x`).

- Inside the loop, a random song is selected using the `np.random.choice()` function from the `list_of_songs`.
- If the selected song has not been played (checked using the `played` array), the script requests the user to continue to play the song. If the user enters 'c', the script proceeds to play the audio using the `play_mp4_audio()` function. Else, the script skips the song to be played and goes to the next iteration.
- After playing a song, the counter variables are updated, and if all songs have been played (`c == 20`), the `played` array is reset, and the counters are reset for the next round.
- The loop continues until the desired number of rounds is completed.



```
/codes$ python3 audplayer.py
Enter the number of times to play the playlist: 2
The next song on queue is aud19. Press 'c' to continue, 's' to skip: c
Now playing aud19.mp4...
The next song on queue is aud15. Press 'c' to continue, 's' to skip: s
The next song on queue is aud11. Press 'c' to continue, 's' to skip: s
The next song on queue is aud17. Press 'c' to continue, 's' to skip: c
Now playing aud17.mp4...
```

Figure 1: output

4 Results

The provided code has been tested and successfully implements a random audio playlist player using the MoviePy library. The script allows users to create a playlist of audio files and play them.