Project Report: Audio Extraction and Shuffling

Sudarshan Shivashankar

May 18, 2023

1 Introduction

The purpose of this project is to extract audio from a series of video files and play them in a shuffled order. The project utilizes the MoviePy library to extract audio from video files and the NumPy library to shuffle the order of the audio files. The extracted audio files are saved in WAV format and played using the 'aplay' command line utility.

2 Objectives

The main objectives of this project are as follows:

1. Extract audio from a series of video files using the MoviePy library.
2. Utilize the NumPy library to shuffle the order of the audio files.
3. Save the extracted audio files in WAV format for playback.
4. Implement a loop that plays the shuffled audio files until the user chooses to stop.
5. Provide documentation and clear instructions for the user to understand and use the program effectively.

3 Methodology

The project was implemented using the following steps:

3.1 Audio Extraction

The program uses the MoviePy library to extract audio from a series of video files. The `extract_audio` function takes the path of the video file and the desired output file path as input. It extracts the audio from the video and saves it as a WAV file.

3.2 Array Shuffling

A list of numbers from 0 to 19 is created. This list is converted to a NumPy array using np.array(). The np.random.shuffle() function shuffles the array in-place, changing the order of the numbers randomly.

3.3 Audio Playback

The shuffled array is converted back to a list using .tolist(). The program then loops through the shuffled numbers and performs the following steps for each number:

1. Prints the current number.
2. Constructs the file paths for the input video file (mp4_file_path) and the output audio file (output_file_path).
3. Calls the extract_audio function to extract the audio from the video and save it as a WAV file.
4. Uses the subprocess.call() function to play the extracted audio file using the 'aplay' command line utility.
5. Prompts the user to enter 'q' if they want to stop playing the songs. If 'q' is entered, the loop breaks; otherwise, it continues.

4 Results

The program successfully extracts audio from a series of video files and shuffles the order of the audio files. The shuffled audio files are played one by one using the 'aplay' command line utility. The user can choose to stop playing the songs by entering 'q' when prompted.

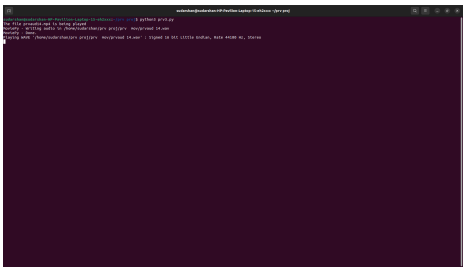

Figure 1: Playing a random song.

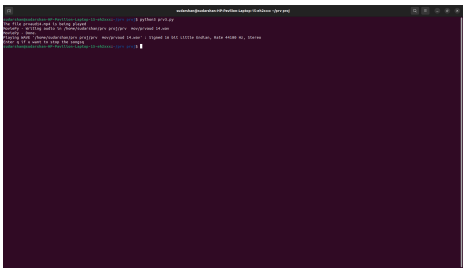

Figure 2: Displaying option to quit the playlist

5 Conclusion

The project accomplished the goal of extracting audio from video files, shuffling the order of the audio files, and playing them using the 'aplay' utility. The program provides a simple and interactive way to enjoy shuffled audio playback. Users can modify the program to work with their own collection of video files and enjoy a randomized audio experience.

6 Future Enhancements

1. Implement a graphical user interface (GUI) for a more user-friendly experience.
2. Add support for additional audio formats and playback options.
3. Integrate with online music platforms for automatic extraction and shuffling of audio files.
4. Provide options for customizing the shuffling algorithm and playback settings.

7 References

1. MoviePy documentation: https://zulko.github.io/moviepy/
2. NumPy documentation: https://numpy.org/doc/
3. subprocess module documentation: https://docs.python.org/3/library/subprocess.html