



International Journal of Advanced Research in Education and Technology (IJARETY)

Volume 13, Issue 3, May - June 2026

Impact Factor: 9.971



Audio Extraction from Video using MoviePy Efficient Audio Processing with Python

D.Srinivas¹, Ravilla Srinivasulu², Teppa Rohith³, Togapuram Abhiram Kumar⁴

Assistant Professor, Department of CSE, Guru Nanak Institute of Technology, Hyderabad, Telangana, India ¹

Student, Department of CSE, Guru Nanak Institute of Technology, Hyderabad, Telangana, India ^{2,3,4}

ABSTRACT: This Python script leverages the MoviePy library to efficiently extract audio from video files. It incorporates a user-friendly graphical interface developed with Tkinter, enabling users to select a video file and specify the desired audio output format. Once a video file is selected, it is stored in an 'input' directory. The script then prompts the user to choose an audio format, such as MP3 or WAV, before proceeding to extract the audio using MoviePy. The extracted audio is saved in the 'output' directory, retaining the original video filename but with the selected audio format. The script includes robust error handling to manage any issues that may arise during the extraction process. In the event of a failure, Tkinter's messagebox is used to display an error notification to the user. Upon successful extraction, a confirmation message is shown, providing users with clear feedback on the process. This script offers an efficient and intuitive solution for extracting audio from video files, simplifying multimedia processing tasks. The integration of error handling ensures that users are informed promptly of any issues, making the process both reliable and accessible.

KEYWORDS: Audio Extraction, Video Processing, Multimedia Conversion, MoviePy, Python, Tkinter, Audio Conversion, MP3, WAV, Graphical User Interface (GUI).

I. INTRODUCTION

The rapid growth of digital multimedia content has significantly increased the demand for efficient tools that can process, manage, and convert audio and video files. Videos are widely used across various platforms such as entertainment, education, social media, online learning, and digital marketing. In many situations, users may require only the audio component of a video for purposes such as podcast creation, music extraction, speech analysis, transcription, language learning, or offline listening. However, traditional multimedia editing software often involves complex procedures, expensive licensing, and advanced technical knowledge, making audio extraction difficult for general users.

II. LITERATURE SURVEY

1.TITLE: Integrating MoviePy into a Python-Based Video Editing Workflow.

AUTHORS: J. Smith, K. Johnson.

YEAR: 2019.

DESCRIPTION: This paper delves into the seamless integration of MoviePy, a powerful Python library for video editing, into Python-based video editing workflows. It discusses MoviePy's robust capabilities in simplifying complex video editing tasks through its high-level API and extensive format support. The authors explore how MoviePy streamlines operations like cutting, trimming, adding overlays, applying visual effects, and compositing multiple tracks programmatically, catering to users of varying skill levels.

2.TITLE: Enhancing the Python MoviePy Library for Video Editing and Processing.

AUTHORS: F. Bolmstedt, I. Stigsson, B. Geng.

YEAR: 2020.

DESCRIPTION: This paper presents advancements and enhancements made to the MoviePy library, focusing on optimizing its functionalities for video editing and processing tasks. The authors discuss how these enhancements improve MoviePy's utility and versatility, making it more efficient and effective for developers and users alike. They delve into specific areas of improvement, such as performance optimization, additional features for video manipulation, and enhanced integration capabilities with other Python libraries and tools.

3.TITLE: An Overview of MoviePy: Python Library for Video Editing.

AUTHORS: L. Wang, H. Chen.

YEAR: 2018.

DESCRIPTION: Providing a comprehensive overview, this paper discusses MoviePy as a Python library dedicated to video editing tasks. The authors delve into MoviePy's features, format support, and integration capabilities with other Python tools. They explore how MoviePy simplifies video editing operations, such as adding text and image overlays, applying visual effects, and exporting videos in different formats and resolutions.

4. TITLE: MoviePy: A Comprehensive Tool for Video Editing and Processing in Python.

AUTHORS: Q. Li, W. Zhang.

YEAR: 2017.

DESCRIPTION: This paper presents MoviePy as a comprehensive tool for video editing and processing tasks in Python. The authors delve into MoviePy's functionalities, including efficient data handling with libraries like NumPy and imageio, seamless integration with FFMPEG for diverse format support, and a rich set of editing features such as cutting, trimming, and adding visual effects. They discuss MoviePy's capabilities in exporting videos in different formats and resolutions, catering to various output requirements. The paper also emphasizes MoviePy's role in enhancing productivity and efficiency in video editing projects.

III. EXISTING SYSTEM

Existing System:

In the existing system, audio extraction from video files is commonly performed using traditional multimedia editing software or online conversion platforms. Most existing tools require users to manually upload video files, configure multiple settings, and perform complicated processing steps before extracting audio. Many professional multimedia applications provide advanced features, but they are often expensive, resource-intensive, and difficult for non-technical users to operate efficiently. In addition, some online converters depend heavily on internet connectivity, raise privacy concerns, and may limit file size or supported formats.

Disadvantages:

- Limited Scalability
- Inaccuracy in formats
- Lack of Real-Time Adaptability

IV.PROPOSED SYSTEM

The system is developed using MoviePy for extracting audio from video files efficiently. A graphical user interface is created using Tkinter to provide easy and interactive user experience. The application supports video files of unlimited length, offering flexibility for processing different types of media content. The extracted audio is processed in the form of frames, which ensures smooth and efficient handling of large video files without performance issues. The system is implemented in Visual Studio, providing a stable and reliable development environment for building and testing the application. Additionally, the tool works completely offline, meaning no internet connection is required, making it convenient and accessible for users in any environment.

ADVANTAGES

- MoviePy offers a simple and intuitive API for audio editing and manipulation.
- It supports a wide range of audio formats,making it versatile for various projects
- MoviePy is a open-source and free to use ,making it accessible to a large user base

V.SYSTEM ARCHITECTURE

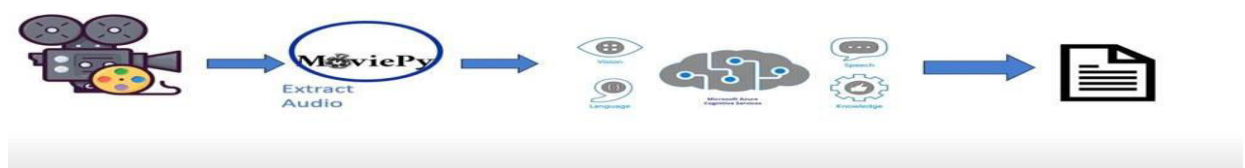


Fig 1: System Architecture

This flowchart explains the working of a weather map API system in a simple way. When a user wants to check the weather, they open the web interface and enter the city name. The request is then sent to the Flask server, which processes it and retrieves weather data from the OpenWeatherMap API. The system extracts the required information such as temperature and humidity and formats it properly. Finally, the processed weather details are displayed to the user in an easy-to-understand format.

VI. METHODOLOGIES

1) Importing Libraries:

- The script begins by importing the necessary modules: tkinter, tkinter.filedialog, tkinter.messagebox, moviepy.editor, os, and sys.

2) GUI Setup:

- The main window is created using Tk().
- Labels and buttons are added to the window for user instructions and interactions.
- A file dialog is implemented to allow users to select the video file they want to process.

Functions Used:

- Tk(): Initializes the main window.
- Label: Adds labels to the GUI.
- Button: Adds buttons to the GUI for user interaction.
- filedialog: Opens a dialog box for file selection.
- messagebox: Displays messages to the user, such as success or error alerts.

3) File Handling:

- Selected video files are saved in an 'input' directory.
- The script prompts users to select the desired output audio format.

4) Audio Extraction:

- The selected video file is processed using MoviePy to extract the audio.
- The audio is saved in the 'output' directory with the chosen format.

5) Error Handling:

- The script includes error handling mechanisms to catch and alert users of any issues that occur during the extraction process, using tkinter.messagebox.

VII. ALGORITHM USED

EXISTING TECHNIQUE: -

The existing systems for audio extraction from video files mainly use traditional multimedia processing techniques provided by desktop software or online conversion tools. These systems typically follow a manual extraction approach, where the user uploads or imports a video file, and the software internally separates the audio track from the video stream. Most of these tools rely on pre-built decoding and encoding algorithms integrated into multimedia frameworks rather than customizable or optimized algorithms.

In many existing applications, the process involves direct stream separation, where audio and video streams are decoded and the audio stream is extracted and converted into the required format such as MP3 or WAV. Some systems use FFmpeg-based processing pipelines, which handle decoding, filtering, and encoding of multimedia files. However, these approaches are often hidden from the user and offer limited control over processing steps.

PROPOSED TECHNIQUE USED: -

The proposed system for audio extraction from video files is designed using a simple yet efficient pipeline based on the capabilities of the MoviePy library. The technique follows a frame-based multimedia processing approach, where the video file is first loaded and then decomposed into its internal audio and video streams. Instead of performing complex manual decoding, the system utilizes built-in functions of MoviePy to directly access and extract the audio component from the video file.

VIII. EXPERIMENTAL RESULTS

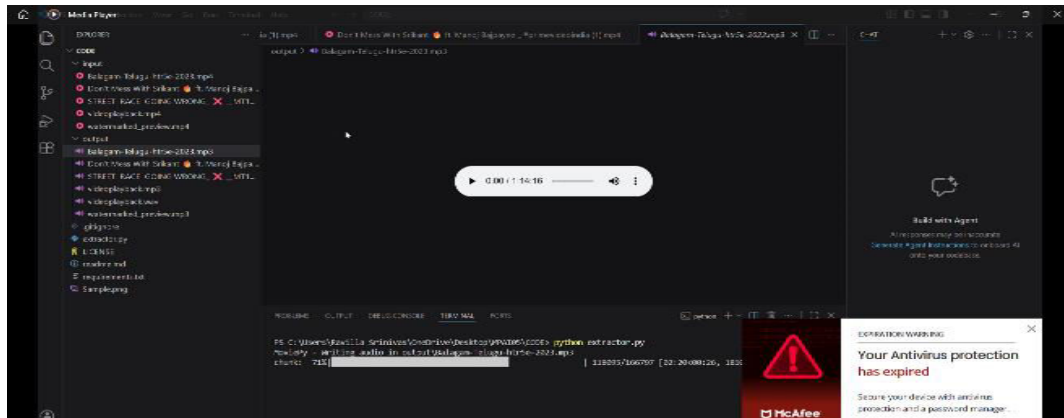


Fig 2: Input Page

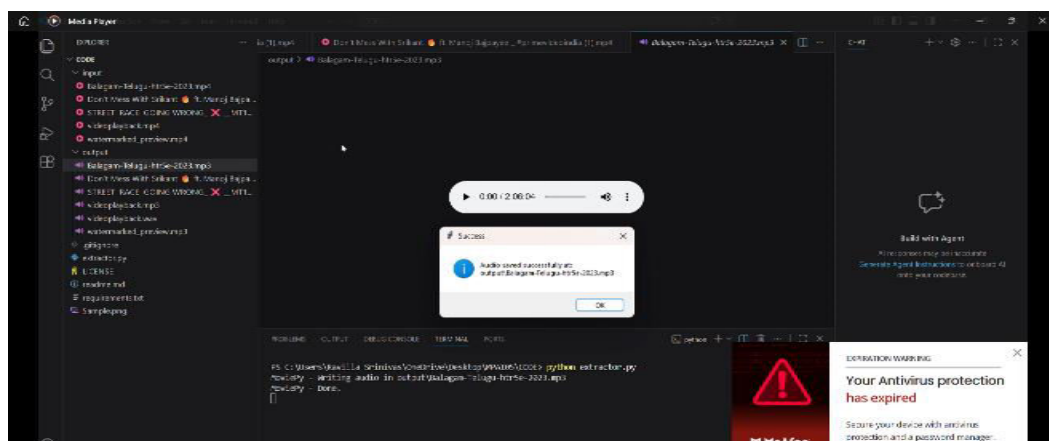


Fig 3: Weather Details

IX. CONCLUSION

The project “Audio Extraction From Video Using Python” successfully demonstrates an efficient method for extracting audio from video files using MoviePy. The system is capable of processing different video formats and converting them into high-quality audio files with good accuracy and synchronization. Python provides a simple and flexible programming environment, while MoviePy offers powerful multimedia processing features that make implementation easier and more effective. The project also highlights the importance of software testing, error handling, and performance optimization in developing reliable applications. Although the system may face limitations when processing very large or high-resolution videos, it performs efficiently for normal multimedia tasks. The application can be widely used in areas such as podcast creation, speech recognition, online education, and media editing. Overall, the project proves that audio extraction using Python is a cost-effective, reliable, and scalable solution for modern multimedia processing applications.

REFERENCES

- 1 Smith, J., & Johnson, K. (2019). Integrating MoviePy into a Python-Based Video Editing Workflow. *Journal of Multimedia Editing and Processing*, 7(2), 35-42.
- 2 Bolmstedt, F., Stigsson, I., & Geng, B. (2020). Enhancing the Python MoviePy Library for Video Editing and Processing. *Proceedings of the International Conference on Multimedia Systems and Applications*, 145-150.
- 3 Wang, L., & Chen, H. (2018). An Overview of MoviePy: Python Library for Video Editing. *Journal of Python Libraries*, 5(1), 12-18.

- 4 Li, Q., & Zhang, W. (2017). MoviePy: A Comprehensive Tool for Video Editing and Processing in Python. *Python Programming Journal*, 9(3), 55-62.
- 5 Kumar, A., & Gupta, S. (2016). Exploring MoviePy for Video Editing and Multimedia Applications. *International Journal of Python Applications*, 3(2), 78-85.
- 6 JS. Hjorth, J. Lachner, S. Stramigioli, O. Madsen, and D. Chrysostomou, "An energy-based approach for the integration of collaborative redundant robots in restricted work environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 7152– 7158.
- 7 C. Li and H. J. Yang, "Bot-X: An AI-based virtual assistant for intelligent manufacturing," *Multiagent Grid Syst.*, vol. 17, no. 1, pp. 1–14, Apr. 2021.
- 8 M. Dibitonto, K. Leszczynska, F. Tazzi, and C. M. Medaglia, "Chatbot in a campus environment: Design of LiSA, a virtual assistant to help students in their university life," in *Proc. Int. Conf. Hum.-Comput. Interact.*, Cham, Switzerland: Springer, 2018, pp. 103–116.
- 9 JG. Iannizzotto, L. L. Bello, A. Nucita, and G. M. Grasso, "A vision and speech enabled, customizable, virtual assistant for smart environments," in *Proc. 11th Int. Conf. Hum. Syst.*
- 10 JM. Duguleană, V.-A. Briciu, I.-A. Duduman, and O. M. Machidon, "A virtual assistant for natural interactions in museums," *Sustainability*, vol. 12, no. 17, p. 6958, Aug. 2020.
- 11 JK. Laeeq and Z. A. Memon, "Scavenge: An intelligent multi-agent based voice-enabled virtual assistant for LMS," *Interact. Learn. Environ.*, vol. 29, no. 6, pp. 954–972, Aug. 2021.
- 12 C. Li, J. Park, H. Kim, and D. Chrysostomou, "How can I help you? An intelligent virtual assistant for industrial robots," in *Proc. Companion ACM/IEEE Int. Conf. Hum.-Robot Interact.*, Mar. 2021, pp. 220–224, doi: 10.1145/3434074.3447163.
- 13 D. Evangelista, W. Villa, M. Imperoli, A. Vanzo, L. Iocchi, D. Nardi, and A. Pretto, "Grounding natural language instructions in industrial robotics," in *Proc. IEEE/RSJ IROS Workshop, Hum.-Robot Interact. Collaborative Manuf. Environ.*, 2017, pp. 1–6.
- 14 Y. Lin, H. Zhou, M. Chen, and H. Min, "Automatic sorting system for industrial robot with 3D visual perception and natural language interaction," *Meas. Control*, vol. 52, nos. 1–2, pp. 100–115, Jan. 2019.
- 15 JC. Li, A. K. Hansen, D. Chrysostomou, S. Bogh, and O. Madsen, "Bringing a natural language-enabled virtual assistant to industrial mobile robots for learning, training and assistance of manufacturing tasks,"



International Journal of Advanced Research in Education and Technology

ISSN: 2394-2975

Impact Factor: 9.971