

**Mini Project Report on**



**Hand Gesture-Based Video Control System**



**Submitted in partial fulfilment of the requirement for the award of the degree of**

**BACHELOR OF TECHNOLOGY IN**

**COMPUTER SCIENCE & ENGINEERING**

**Submitted by:**

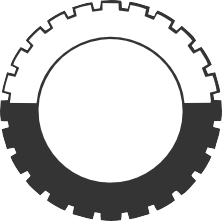
**Student Name: Mimanshu Negi University Roll No.: 2118791**

***Under the Mentorship of***

**Ms. Himadri Vaidya**

**Department of Computer Science and Engineering Graphic Era Hill University**

**Dehradun, Uttarakhand**



**CANDIDATE’S DECLARATION**

I hereby certify that the work which is being presented in the project report entitled **“Sorting Visualizer using html, CSS and JavaScript”** in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering of the Graphic Era Hill University, Dehradun shall be carried out by myself under the mentorship of **Ms. Himadri Vaidya**, Department of Computer Science and Engineering, Graphic Era Hill University, Dehradun.

**Name: Mimanshu Negi University Roll no.: 2118791**

**Hand Gesture-Based Video Control System**

**1. Introduction**

In an increasingly digital world, the demand for more intuitive and contact-free methods of interacting with devices is growing. This project aims to develop a hand gesture recognition system to control video playback functions such as play, pause, volume adjustment, and navigation using a webcam. By leveraging computer vision and machine learning techniques, this system provides a natural and user-friendly interface for media control.

**2. Problem Statement**

The primary goal of this project is to design a system that can recognize specific hand gestures and map them to video control functionalities. The system captures real-time video through a webcam, detects hand gestures, and controls video playback without requiring physical contact. This offers a seamless and intuitive user experience, enhancing accessibility and convenience.

**3. Motivation**

The motivation behind this project is to enhance user experience by providing a touch-free method of video control. This is particularly beneficial in scenarios where hands-free operation is preferred, such as during cooking, giving presentations, or for individuals with physical disabilities. Additionally, this project explores the practical applications of computer vision and machine learning technologies, showcasing their potential in everyday tasks.

**Benefits:**

* **Accessibility:** Assists individuals with physical disabilities.
* **Hygiene:** Reduces the need for physical contact, especially useful in public or shared spaces.
* **Convenience:** Enhances user experience by providing a more natural way to interact with media.

**4. Tools Used**

4.1 Programming Language

Python: Chosen for its extensive libraries and ease of use.

4.2 Libraries

* OpenCV (cv2): For image and video processing. OpenCV is a powerful library for computer vision tasks, allowing for real-time image capture and processing.
* MediaPipe (mediapipe): For hand landmark detection. MediaPipe provides high-fidelity hand tracking, making it ideal for gesture recognition applications.
* IPython Widgets (ipywidgets): For creating interactive widgets. These widgets enhance the user interface, allowing for real-time updates and interactions.
* Threading (threading): For handling concurrent execution. Threading ensures the gesture detection process runs smoothly alongside other operations.
* IPython Display (IPython.display): For displaying widgets and executing JavaScript, integrating video controls directly into the environment.

4.3 Hardware

Webcam: A standard webcam is used for capturing real-time video input, essential for gesture detection.

**5. Methodology**

5.1 Initialization

The project begins by importing the necessary libraries and initializing MediaPipe for hand detection. The HandGestureRecognition class is defined to handle gesture detection. This class initializes MediaPipe's hands solution with specified detection and tracking confidence levels, ensuring accurate and reliable hand tracking.

5.2 Gesture Detection

The system captures video frames from the webcam and converts them to the RGB color space, as MediaPipe operates in this format. Using MediaPipe, it detects hand landmarks in these frames. Specific gestures are identified based on the positions of these landmarks.

Key Gestures:

* Open Palm Gesture ("play"): Recognized when all fingers are extended.
* Closed Fist Gesture ("pause"): Recognized when all fingers are curled.
* Volume Up Gesture ("volume\_up"): Detected when the index and middle fingers are up, and the ring and pinky fingers are down.
* Volume Down Gesture ("volume\_down"): Detected when the index finger is up, and the other fingers are down.
* Sliding Detection: Recognizes when the index finger and thumb are close together and detects the direction of sliding based on movement.

5.3 Video Control

A VideoPlayer class is created to handle video playback. This class uses HTML and JavaScript to embed and control the video player within a Jupyter Notebook. The player responds to the detected gestures, enabling functionalities like play, pause, volume adjustment, and navigation.

**Functionalities:**

* Play/Pause: Controlled by open palm and closed fist gestures.
* Volume Adjustment: Controlled by volume up and volume down gestures.
* Navigation: Moving ahead or behind in the video based on sliding gestures.

5.4 Execution

The video player is initialized with a YouTube video URL. A separate thread is started to continuously detect gestures and control the video in real-time. This setup ensures that the gesture detection process does not block the main execution thread, allowing for smooth and uninterrupted video playback.

**6. Results**

The system successfully recognizes hand gestures and maps them to video control functions. The detection accuracy and response time are satisfactory for real-time interaction. The video control functionalities (play, pause, volume adjustments, navigation) operate as expected based on the detected gestures.

6.1 Accuracy

The system demonstrates high accuracy in detecting hand gestures under various lighting conditions and backgrounds. It consistently identifies gestures with minimal false positives or negatives.

6.2 User Feedback

Initial user feedback indicates a positive reception, highlighting the system's ease of use and responsiveness. Users appreciate the convenience and novelty of controlling video playback through hand gestures.

6.3 Limitations

Despite its success, the system has some limitations:

Lighting Sensitivity: Performance may degrade in poor lighting conditions.

Gesture Complexity: Complex or ambiguous gestures may not be accurately detected.

**7. Conclusion**

This project demonstrates the feasibility of using hand gesture recognition for touch-free video control. It provides a user-friendly interface that can be beneficial in various applications. Future work could include improving the robustness of gesture detection, expanding the system to support additional gestures and functionalities, and exploring applications beyond video control.

7.1 Future Work

Enhanced Gesture Set: Incorporate additional gestures for more control options.

Improved Detection: Optimize the system for varying lighting conditions and backgrounds.

Application Expansion: Apply the technology to other areas, such as smart home control or interactive presentations.

**8. References**

OpenCV: https://opencv.org/

MediaPipe: https://mediapipe.dev/

IPython Widgets: https://ipywidgets.readthedocs.io/en/latest/