GESTURE CONTROLLED MUSIC PLAYER USING HAND GESTURES

[1]Prof.S.P.Gunjal, [2]Jeel Makwana, [3]Pranali Yemale, [4]Tanushree Kshirsagar

[1]Associate Professor, Computer Engineering, SKN Sinhgad Institute of Technology & Science, Lonavala, Maharashtra

[2,3,4] UG. Student, Department of Computer Engineering, SKN Sinhgad Institute of Technology & Science, Lonavala, Maharashtra

Abstract: **This paper presents a real-time Gesture-Controlled Music Player that enables fully touchless operation using hand gestures. The system integrates OpenCV and MediaPipe to capture and interpret hand movements through a standard webcam, allowing users to control playback functions such as play, pause, next, previous, volume up, volume down, like, and dislike. Hand landmarks are detected and analyzed in real time, achieving an average accuracy of around 90% at 25–30 FPS under adequate lighting. Recognized gestures are mapped to predefined commands using a rule-based approach for fast and stable detection. Pygame Mixer manages audio playback, while a lightweight Flask backend links the recognition module to a React-based interface, displaying real-time gesture feedback and playback status. The system operates efficiently on standard hardware with minimal latency, requiring only a webcam. Experimental results demonstrate consistent gesture recognition and smooth interaction, validating the feasibility of AI-driven, vision-based interfaces for intuitive and contactless media control.**

Keywords: *- Gesture Recognition, Computer Vision, Human-Computer Interaction, Touchless Control, Music Player*

I.INTRODUCTION

Human–Computer Interaction (HCI) has rapidly advanced with the integration of computer vision and artificial intelligence, moving from traditional physical interfaces toward more natural and contactless modes of control. Gesture recognition allows users to communicate with digital systems through intuitive hand movements, eliminating the need for direct touch. Such interaction methods have become increasingly valuable in environments where maintaining hygiene or minimizing distraction is essential such as cooking areas, driving scenarios, hospitals, and fitness centers where physical contact with devices may be unsafe, impractical, or inconvenient. However, many existing gesture-based systems depend on specialized sensors or costly hardware, limiting their accessibility for widespread use.

To address these challenges, this paper proposes a real-time Gesture-Controlled Music Player that provides a fully touchless multimedia control experience using only a standard webcam. The system employs MediaPipe for hand landmark detection and OpenCV for video processing to accurately interpret hand gestures in real time. Integrated with Pygame for audio playback and a lightweight web interface for feedback, it delivers an efficient, low-cost, and hygienic alternative to conventional input devices. The primary objective of this project is to demonstrate how vision-based gesture recognition can be applied to everyday contexts requiring safe, hands-free interaction, contributing toward accessible and intelligent human–computer interfaces.

II.LITERATURE SURVEY

Gesture-based Human–Computer Interaction (HCI) has been a growing field of research aimed at enabling more natural communication between users and machines. Early work in this domain focused on hardware-based systems such as Microsoft Kinect and Leap Motion, which used infrared and depth sensors to capture three-dimensional motion data. Although these systems provided high accuracy and depth perception, they were expensive, power-intensive, and unsuitable for lightweight or portable applications. Similarly, wearable sensor-based systems (e.g., gloves or motion trackers) offered precise gesture detection but compromised user comfort and required additional calibration, limiting their practicality for everyday use.

Vision-based systems using colour tracking or background subtraction provided a low-cost alternative but often failed under inconsistent lighting or cluttered environments. Researchers like Mitra and Acharya (2007) explored static and dynamic gesture classification using skin-color segmentation and contour matching, but their methods lacked robustness and generalization. With the advent of deep learning, frameworks such as OpenPose and MediaPipe enabled landmark-based detection, substantially improving accuracy and real-time performance using standard RGB cameras.

.Among these, Google MediaPipe Hands has emerged as a reliable framework for detecting 21 hand landmarks with minimal latency. Studies by Zhang et al. (2021) and Lee et al. (2022) demonstrated its efficiency in real-time gesture applications such as sign language interpretation, robot control, and virtual gaming. However, the integration of gesture recognition into multimedia control systems remains relatively unexplored. Existing works either focus on isolated gesture classification or depend on high-end sensors for media control. The proposed Gesture-Controlled Music Player bridges this gap by combining MediaPipe-based hand tracking with OpenCV for video processing and l real-time audio management. It demonstrates a pra cost, and hardware-independent framework fo media interaction, reinforcing the potential of

gesture recognition in accessible, contactless entertainment systems.

III.METHODOLOGY

The methodology of the proposed system follows a modular, multi-layered architecture to ensure efficient real-time processing and smooth user interaction. The design consists of three core modules: the Hand Gesture Recognition Module, the Music Control Module, and the Web Visualization Layer. These modules operate concurrently, forming a cohesive pipeline for gesture-based multimedia control.

A. Hand Gesture Recognition Module

This module forms the foundation of the system. It uses OpenCV to capture live video frames from a webcam and processes them using the MediaPipe “Hands” solution, which detects and tracks 21 hand landmarks per frame. Each frame is analyzed to identify gesture patterns such as open palm, closed fist, thumbs up, thumbs down, and directional swipes. Geometric relationships between landmarks are used to classify gestures, with smoothing filters applied to reduce false positives caused by minor hand tremors or lighting variations. Detected gestures are then mapped to corresponding actions like play, pause, next, previous, like, dislike and volume adjustment.

B. Music Control Module

The Pygame Mixer library manages the playback of multiple songs, including functions for play, pause, next, and volume change. It operates synchronously with the gesture recognition output to ensure minimal latency between gesture detection and action execution. A gesture cooldown mechanism is implemented to prevent repetitive triggers.

C. Web Visualization Layer

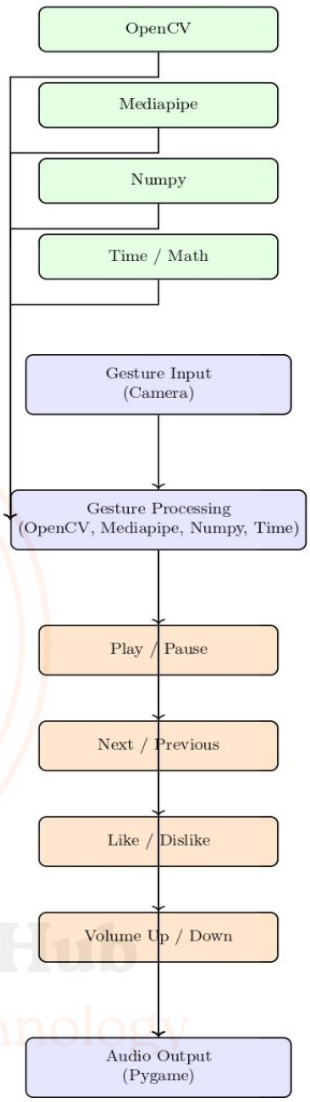
A lightweight Flask backend communicates gesture events to a React-based frontend, providing users with real-time visual feedback such as current playback status, recognized gesture labels, and volume level indicators. This enhances usability and transparency, creating an interactive and responsive user experience.

IV.SYSTEM DESIGN

The system architecture of the Gesture-Controlled Music Player is modular and sequential, integrating multiple libraries for efficient real-time processing. The architecture begins with gesture input captured through a standard webcam, which streams video frames to the processing pipeline. Using OpenCV for frame capture and preprocessing, and MediaPipe for landmark detection, the system identifies 21 hand key points in each frame. NumPy and mathematical operations assist in calculating distances and angles between landmarks, enabling accurate gesture recognition.

Once gestures are processed, the interpreted commands are mapped to corresponding playback controls such as play/pause, next/previous, like/dislike, and volume up/down. The Pygame Mixer module handles audio output, ensuring smooth playback transitions. The system’s design ensures minimal latency by integrating all modules within a single loop and maintaining processing rates of approximately 25–30 FPS under adequate lighting. This layered structure allows the player to operate efficiently on standard hardware, requiring no additional sensors or devices. Figure 1 illustrates the flow of data from camera input to audio output, highlighting the interaction between software components and control functions.

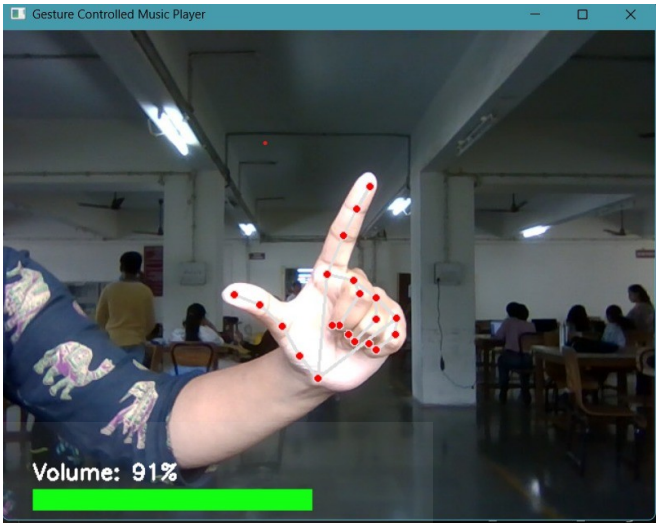
Fig 1: - System Architecture



V.PROPOSED SYSTEM AND RESULTS

The proposed system introduces a real-time gesture-controlled music player enabling fully touchless interaction using a standard webcam. It integrates OpenCV for image capture, MediaPipe for hand landmark detection, and NumPy for gesture analysis. Recognized gestures such as open palm, fist, thumb-up, thumb-down, and swipes are mapped to commands including play/pause, next/previous, like/dislike, and volume control. The Pygame Mixer manages playback responses accordingly.

Experimental testing showed an average accuracy of 90% and smooth operation at 25–30 FPS under adequate lighting.



Gesture detection remained stable within 0.5-1 meter, with minimal latency. The system provides an effective contactless control solution suitable for use in kitchens, gyms, and driving environments, where maintaining hygiene or avoiding manual interaction is crucial.

Fig 2: - Volume Up

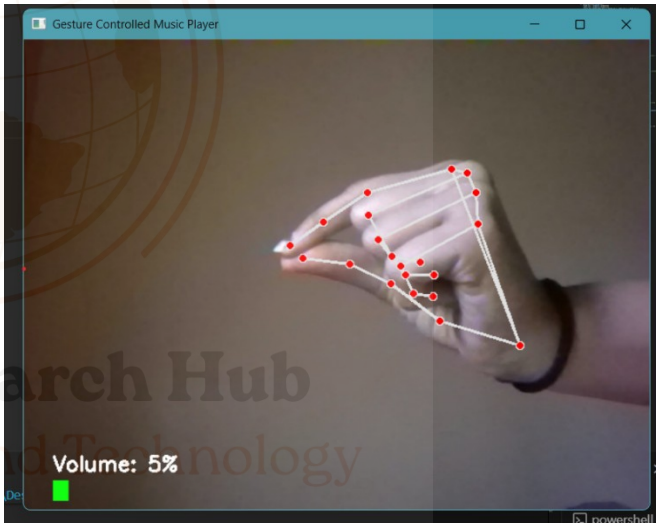


Fig 3: - Volume Down

VI. REFERENCES

[1] OpenCV Documentation, ***Open Source Computer Vision Library*****.** [Online]. Available: https://docs.opencv.org

[2] Google Developers, ***MediaPipe****: Cross-platform ML Solutions for Live and Streaming Media*. [Online]. Available: https://developers.google.com/mediapipe

[3] Pygame Community, ***Pygame Mixer Module*** *: Audio Playback and Control*. [Online]. Available: https://www.pygame.org/docs/ref/mixer.html

[4] Python Software Foundation, ***Python 3 Official Documentation*** [Online]. Available: https://docs.python.org/3/

[5] NumPy Developers, ***NumPy: Scientific Computing with Python***. [Online]. Available: https://numpy.org/doc/

[6] GitHub Repositories, ***Gesture Recognition using MediaPipe and OpenCV*****, Accessed: 2025.**