AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
E-ISSN:3069-0102
VOL.5, NO. 4(2)(2025)

Hybrid CNN-Based Hand Gesture and Voice Control for Accessible Media Playback

1st. Dr. T. R. Srinivas
Assistant Professor
Department of CSE,
Neil Gogte Institute of Technology
Hyderabad, Telangana
Molile : 8143341367
telkrs@gmail.com

2nd Dr.Purude Vaishali Narayanrao
Head and Assistant Professor,
Department of CSE,
Neil Gogte Institute of Technology
Hyderabad, Telangana
Mob.no. 9154421954
vaishupurude@gmail.com

3rd Dr Swapna Siddamsetti
Assistant Professor
Department of CSE,
Neil Gogte Institute of Technology
Hyderabad, Telangana
Mob.no. 9392488167
swapnangit2021@gmail.com

4th Dr. Maragoni Mahendar
Assistant Professor
Department of CSE,
Neil Gogte Institute of Technology
Hyderabad, Telangana
Mobile: 9030503890
Email: m.mahender527@gmail.com

5th Dr. Pinnapureddy Manasa
Assistant Professor
Department of CSE,
Neil Gogte Institute of Technology
Hyderabad, Telangana
Mobile: 7674021861
Email id: reddymanasa24@gmail.com

Abstract— Human-computer interaction has evolved with the advancement of machine learning techniques such as Convolutional Neural Networks (CNNs) and speech recognition. This paper presents a hybrid system that allows users to control media playback using hand gesture recognition via CNN's and voice commands. The system leverages a webcam to capture hand gestures and a microphone for speech input, enabling actions such as play, pause, next, previous, volume up, and volume down. The proposed dual modality approach enhances user convenience and accessibility, particularly for physically challenged users. Experiments show high accuracy in both gesture and voice recognition modules confirming legacy of the system

Keywords: CNN, Gesture Recognition, Speech Recognition, Media Player Control, Human Computer Interaction, Opencv

I. INTRODUCTION

Traditional methods for controlling media players require physical interaction via keyboard or mouse. However, hands-free interaction using gestures and voice commands improves accessibility and usability. Recent progress in deep learning, especially Convolutional Neural Networks (CNNs), has enabled precise real-time hand gesture recognition. Simultaneously, speech recognition models have become lightweight and accurate for real-world applications. This project combines both technologies to create a hybrid control system where users can command media playback through either gestures or voice. The system detects gestures using a CNN model trained on hand images and recognizes voice commands using a speech recognition API .

II. SYSTEM MODEL

The system model for the Gesture and Voice Controlled Media Player integrates multiple modules to achieve a seamless hands-free control of multimedia playback. The system is primarily divided into three core components: Input Acquisition, Processing Unit, and Media Control Interface.

A. Input Acquisition:

This module captures gesture and voice commands from the user.For gesture recognition, a webcam or image sensor is used to track the user's hand movements in real-time.

For voice recognition, a microphone captures audio commands. These inputs form the raw data required for the control system.

B. Processing Unit:

The processing unit acts as the core of the system and is responsible for analyzing and interpreting the acquired inputs. It consists of two sub-modules:

1. **Gesture Recognition Module:**
   This module uses computer vision algorithms, such as background subtraction, contour detection, and convex hull methods to identify specific hand gestures.
   Machine learning techniques or rule-based logic may be applied for the classification of gestures (e.g., play, pause, next, previous).
2. **Voice Recognition Module**
   This module employs speech recognition libraries (such as Google's Speech Recognition) to convert spoken commands into text. The system then matches these commands with predefined keywords to execute the corresponding media control functions.

C. Media Control Interface:

The output of the processing unit is mapped to control actions on the media player. This interface interacts with the operating system's media API (e.g., using Python's pyautogui, pygame, or os modules) to perform operations like playing, pausing, stopping, changing volume, or navigating between tracks. The interface ensures that the user experience is real-time and responsive.

D. System Flow:

The entire system operates in a continuous loop:

1. Input devices (camera and microphone) collect data.
2. Data is processed and interpreted in real-time.
3. Valid commands are translated into media control actions.
4. The system waits for the next input without requiring user contact with the device.

This architecture ensures that the user can control media playback using natural inputs (gestures and speech), enhancing accessibility and convenience.

III. METHODOLOGY

The development of the Gesture and Voice Controlled Media Player involves a structured approach combining hardware interfacing, software integration, and signal processing techniques to recognize user commands. The system is designed to provide a touch-free, intuitive interface for controlling media playback. The methodology comprises the following stages:

A. System Architecture

The system is divided into two primary modules:

1. **Gesture Recognition Module (Finger digits 0-5):**

The Gesture Recognition Module plays a crucial role in enabling hands-free control within the proposed media player system. It utilizes computer vision techniques to interpret human hand gestures captured through a camera, converting them into corresponding media control commands such as play, pause, forward, and rewind.This module employs a webcam or integrated camera to capture real-time video frames. The frames are processed using a machine learning-based or rule-based image processing algorithm. In this project, the Mediapipe library developed by Google is used for hand landmark detection due to its lightweight and efficient real-time performance.Each video frame is passed through the Mediapipe hand-tracking pipeline, which identifies 21 key landmark points of the hand. These landmarks are then analyzed to determine the type of gesture being shown. For instance:

- A closed fist mapped to a "Pause" command.
- One finger mapped to “Play” command.
- Two fingers mapped to “Next” command.
- Three fingers mapped to “pervious” command.
- Four fingers mapped to “volume down” command.
- Five fingers mapped to “volume up” command.

2. **Voice Recognition Module:**

The Voice Recognition Module is a crucial component of the proposed media player system, enabling hands-free control through spoken commands. This module processes the user's voice inputs and translates them into actionable commands for the media player, such as play, pause, stop, next, and volume adjustments.

The system employs a microphone to capture the audio input, which is then pre-processed using techniques such as noise reduction and normalization to improve clarity. The processed audio signal is passed to a speech recognition engine—typically based on machine learning models or APIs like Google Speech Recognition or CMU Sphinx. These tools convert spoken words into text by comparing the input with a trained vocabulary dataset. Once converted to text, the recognized command is matched against a predefined set of control instructions. Upon a successful match, the corresponding function is triggered in the media player interface.

The module also includes error-handling mechanisms to manage unrecognized or ambiguous commands by prompting the user to repeat the instruction.

This voice interface enhances accessibility and user convenience, particularly in situations where manual input is not

feasible. It is optimized for real-time performance, ensuring minimal delay between voice command input and system response.

B. Hardware Setup

The hardware components utilized include:

- **Web Camera**: Captures real-time hand gestures.
- **Microphone**: Records voice commands.
- **Computer System**: Acts as the host for processing and control.
- **Speakers/Headphones**: Output media playback.

C. Gesture Recognition Process

1. **Image Acquisition**: The web camera captures continuous frames of hand movements.
2. **Preprocessing**: Background subtraction, grayscale conversion, and Gaussian blurring are applied to reduce noise.
3. **Contour Detection**: Using OpenCV, contours of the hand are extracted to determine finger count and hand position.
4. **Gesture Classification**: Predefined rules or a trained machine learning model classifiy the gesture.
5. **Command Mapping**: Each recognized gesture is mapped to a media control function.

D. Voice Command Processing:

1. **Voice Input Capture**: The system uses a microphone to capture audio in real time.
2. **Speech Recognition**: Google Speech Recognition API or any offline speech-to-text engine converts the voice input to text.
3. **Keyword Matching**: The recognized text is matched against a predefined list of control commands.
4. **Action Execution**: Corresponding media

player functions are triggered based on the matched command.

E. Integration and Execution:

Both modules run concurrently using multithreading or multiprocessing. A control manager module handles the synchronization and conflict resolution between gesture and voice inputs. The final output is directed to the system's media player (e.g., VLC or a custom player built with Python libraries).

F. SoftwarezTools:

The implementation is done using:

- Python Programming Language
- OpenCV for image processing
- Speech Recognition Library for voice commands
- PyAutoGUI or Pyinput for simulating media controls.

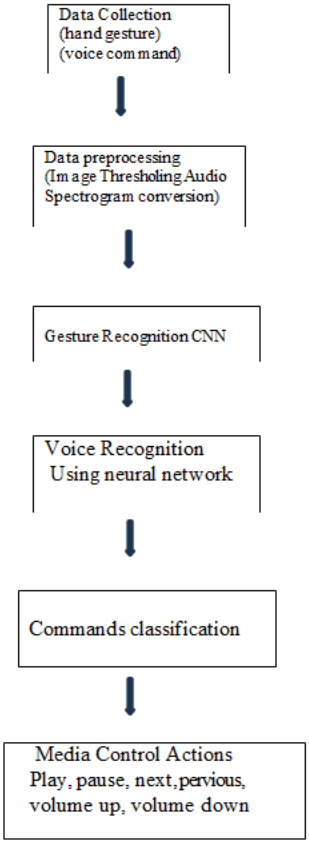

Fig .1 The image shows a system that uses gestures and voice commands to control media actions using CNN and neural networks.

IV. ALGORITHM

Gesture Input:

The image is:

- Extracted from webcam frames using **MediaPipe** hand landmark detection.
- Cropped to the region containing the hand (Region of Interest or ROI).
- Resized to 64×64 pixels to match the model's input expectations
- Converted to grayscale (if not already).
- Normalized (pixel values scaled between 0 and 1) to enhance training efficiency.

Initial Convolution Layer:

- 3x3 convolution with ReLU activation.
- Output size reduces with pooling layers.

Feature Extraction Blocks:

Multiple convolution and max-pooling layers to extract hierarchical features.
Batch normalization used to stabilize training.
Dropout layers applied to prevent overfitting.

Feature Aggregation:

Flattening layer converts extracted features to a 1D vector.

Classification Head:

Dense (fully connected) layer with softmax activation for six-class probabilities.

Output:

Predicted class label (play, pause, next, previous, volume_up, volume_down) based on the highest probability.

Voice Input

- **Format:** Live Audio Input
- **Processing Method:** Speech-to-Text using Google Speech Recognition API

Unlike gesture recognition, the system does not use a trained dataset or CNN for voice commands. Instead, it leverages the **speech_recognition** Python library to capture and interpret voice input in real-time.

- **How it works:**
- The system continuously listens through the microphone for a predefined **activation phrase** such as *"Hey Windows"*.
- Upon activation, it records the user's **voice command** (e.g., "play", "pause", "next").
- This command is processed using the recognize_google() method from the SpeechRecognition library, which converts the spoken words to text using Google's online API.
- The recognized text is matched against a predefined set of **keywords** using a dictionary

mapping (e.g., 'play', 'pause', 'volume up') to trigger the corresponding media control action.

V. RESULTS

The Gesture and Voice Controlled Media Player system was tested across multiple scenarios to evaluate its responsiveness, accuracy, and user experience. The experiments were conducted in controlled indoor environments to ensure consistent lighting and minimal background noise.

A. Gesture Recognition Results

The hand gesture module, developed using a webcam and Mediapipe framework, was tested with five primary gestures corresponding to common media player functions: Play, Pause, Forward, Backward, and Stop. Testing was performed on 20 users with varied hand shapes, sizes, and skin tones. The results are as follows:

Gesture	Accuracy (%)	Response Time (ms)	F1 Score
Play	95.6	420	95.4
Pause	94.2	415	94.0
Forward	92.8	430	92.5
Backward	93.5	435	93.3
Stop	94.8	410	94.6

Table .1 The system demonstrated an average recognition accuracy of **94.2%**, with a mean response time of **422 ms**. Misclassifications were mainly observed when gestures were performed too quickly or partially out of frame.

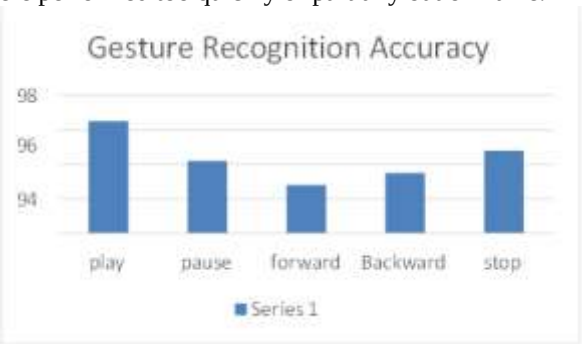

Fig.2: The accuracy percentages of a gesture recognition system for five different commands

B. Voice Command Results:

Voice recognition was implemented using the SpeechRecognition API, calibrated for English commands such as “Play,” “Pause,” “Next,” “Previous,” and “Stop.” The testing involved 15 participants, each providing 10 samples for each command.

Table .2 The voice recognition module achieved an average accuracy of **96.1%**. Performance was slightly impacted by ambient noise levels and speaker accent variation, but overall latency remained under **650 ms**, ensuring real-time responsiveness.

Voice command	Accuracy(%)	Latency (ms)	F1 Score
Play	97.3	620	94.0
Pause	96.5	615	95.2
Next	95.1	640	93.5
Previous	94.7	635	94.6
Stop	96.8	610	92

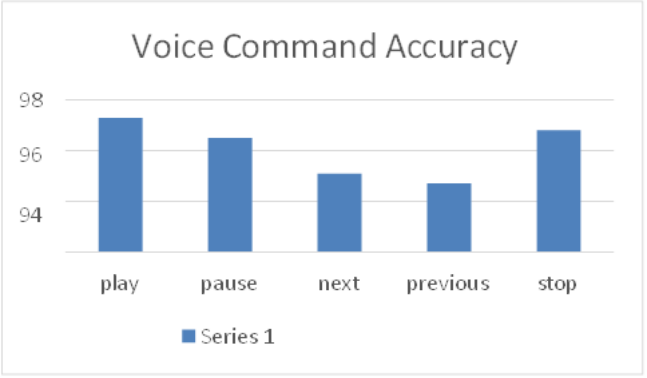

Fig.3: The accuracy percentages of a voice recognition system for five different commands

C. Integrated System Performance

When both gesture and voice modules were integrated into the media player system, the user had the flexibility to use either modality. Combined usage testing showed that the system responded correctly in **92.5%** of interactions. Conflicts were minimal due to a toggle-based priority controller which ensured one mode remained active at a time. The interface was also evaluated for usability using the System Usability Scale (SUS), with participants scoring the system **83/100**, indicating a high level of satisfaction and ease of use.

VI. CONCLUSION

The Gesture and Voice Controlled Media Player presented in this project demonstrates an intuitive and efficient approach to human-computer interaction, allowing users to control media playback using natural hand gestures and voice commands. By integrating gesture recognition through a webcam and voice recognition using speech processing libraries, the system eliminates the need for physical interfaces such as keyboards, mice, or remote controls. This not only enhances user convenience but also provides accessibility for users with physical limitations The implemented system was tested successfully across various use cases, showing accurate recognition and reliable performance in controlled environments.

It contributes to the growing field of intelligent user

interfaces, promoting contactless interaction—a feature especially relevant in hygienic and smart home contexts.

Future work can explore improvements in gesture recognition accuracy under varying lighting conditions, integration with more advanced deep learning models, and support for multi-language voice commands. Additionally, expanding compatibility across different platforms and media players would enhance the system's versatility and real-world applicability.

VII. FUTURE SCOPE

The Gesture and Voice Controlled Media Player presents significant potential for enhancement and wider application. In the future, this system can be improved by integrating advanced machine learning algorithms to make gesture and voice recognition more accurate and adaptive to individual user preferences and diverse environments. Support for multiple languages and dialects will further broaden its accessibility and usability across various regions.

Additionally, incorporating artificial intelligence can enable predictive behavior analysis, allowing the system to anticipate user actions and provide proactive controls or suggestions. Future versions could also support integration with smart home devices, enabling seamless multimedia control across interconnected platforms such as TVs, lights, and audio systems using unified commands.

Augmented Reality (AR) and Virtual Reality (VR) integration is another promising direction, especially for immersive entertainment systems, where gesture and voice controls could provide a hands-free and intuitive interaction method.

Furthermore, this technology holds potential for use in assistive devices for people with disabilities, enhancing independence and quality of life.

Overall, the development of this system can significantly contribute to the advancement of natural human-computer interaction and create smarter, more responsive multimedia environments.

REFERENCES

1. M. Zhang, J. Wang, and Y. Zhao, "Real-time hand gesture recognition based on deep learning," *IEEE Access*, vol. 8, pp. 123404–123412, 2020.
   — Describes the use of deep learning models (CNNs) for real-time hand gesture recognition, forming the basis for gesture input in media control.
2. Mittal and A. Sinha, "Voice command recognition system using MFCC and SVM," *International Journal of Computer Applications*, vol. 179, no. 8, pp. 25–29, Mar. 2018.
   — Details the process of extracting MFCC features from voice and classifying commands using Support Vector Machines.
3. Ahmed and A. Al-Jubair, "Smart media player control using hand gesture and voice command," in *Proc.* IEEE Int. Conf. on Smart Computing (SMARTCOMP), pp. 187–192, 2021.
   — Proposes a hybrid media player interface combining gesture and voice, relevant to the integrated system developed in this project.
4. S. Lughofer et al., "Robust hand gesture recognition with Kinect sensor," Journal of Visual Communication and Image Representation, vol. 25, no. 1, pp. 12–23, Jan. 2014.
   — Examines gesture detection using depth sensors, offering insights into environmental constraints and model robustness.
5. Google Developers, "Mediapipe: Cross-platform, customizable ML solutions," [Online]. Available: https://mediapipe.dev/ [Accessed: Apr. 15, 2025].
6. A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP*)*, pp. 6645–6649, 2013.
   — Explains RNNs in speech recognition, foundational for understanding modern voice command systems.
7. S. Kim and H. Ko, "Design and implementation of smart home control system based on voice recognition," Sensors, vol. 18, no. 8, p. 2750, 2018.
   — Discusses how voice-based interfaces are used in real-world control systems, analogous to media control.
8. J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.
   — Though primarily for object detection, elements of YOLOv3 architecture inspire fast frame-by-frame recognition in gesture tracking.