Grenze International Journal of Engineering and Technology, June Issue

Hand Gesture-Controlled Media Player with Adaptive user Profiles

Shaswat Surya S[1], Shoban Raj TN[2] and S. Janagiraman[3]
[1-2]UG Student, Department of Computer Science and Engineering, St. Joseph's Institute of Technology, OMR, Chennai-119, India
Email: shaswatsurya12@gmail.com, shoban3084@gmail.com
[3]Associate Professor, Department of Computer Science and Engineering, St. Joseph's Institute of Technology, OMR, Chennai-119, India
Email: jani28cse@gmail.com

***Abstract*— The paper presents a gesture-based media control system augmented with adaptive user profiles and the use of computer vision and facial recognition. The gesture control system is developed in Python using OpenCV and MediaPipe so that the user may control multimedia playback without the use of touch, for example through hand gestures. The most unique aspect of the solution is that it provides support for adaptive, multi-user scenarios, through the use of JSON for profile storage. This means that each users' gesture preferences and media use activity is personalized upon facial recognition. The solution framework emphasizes user convenience, hygiene, and inclusivity for differently-abled users or for use in public space scenarios. The paper describes the design methods, implementation modules, and evaluative performance. Our implementation shows that there is a practical and scalable system integrating real-time hand gesture interaction with adaptive behavior using commodity hardware.**

***Indexed Terms*— MediaPipe, media control system, face recognition, OpenCV, adaptive user profile, JSON- based storage.**

I. INTRODUCTION

The development of human-computer interaction (HCI) has enabled the use of intuitive, non-contact interfaces such as gesture-based systems. Remote controls or hand- held input devices are generally in decline and being supplanted by alternative vision-based means of interaction that are more natural and hygienic. The acceleration of the need for non-contact interaction with digital systems has been exacerbated by the onset of the COVID-19 pandemic. We present a real-time hand gesture-based media player controller that allows for adaptive user profiles using the following technologies: MediaPipe for robust hand tracking and gesture identification, OpenCV for face recognition and real-time video feed work, and pyautogui for media control automation. Consistent with intelligent systems, our media controller does not use a fixed set of gesture-to-action mappings; rather, the controller defines and prescribes gesture-to-action mappings for each identified user, by utilizing per-user JSON-based profile configurations data to store or retrieve data. The purpose –to create a user intelligent, inclusive, and specific control environment, that can adapt over time.

Grenze ID: 01.GIJET.12.2.21_21

This paper will detail the rationale for the media controller, the related works in gesture and adaptive systems, and the respective implementation modules which result in the presented performance results.
The gap found in user dynamic applications and pre- defined rigid conceptions of gesture recognition interfaces can be solved by customizing assigned gesture layouts based on facial recognition. Our system improves existing technologies by tailoring personalization modules through gesture mapping while enabling face-recognition based user identification. This allows the user interface to be customized to meet each user's specific preferences and be responsive to their surroundings. Such advanced systems have the potential to support users with disabilities or provide new aids to allow wider public access to kiosks through intuitive gesture-driven controls. With a plug-and-play concept, users can share systems like computers, tablets, or smartphones in flexible environments where multiple people can access and use the devices simultaneously. This promotes ease of customization while actioning dynamic configuration for face-based gesture assigned interfaces responsive to configurable parameters in real-time. Such modular paradigms improve system performance and lessen troubleshooting complexities, allowing format freedom beyond gesture-activated media selection.

II. Literature Review

The development of gesture recognition systems for human-computer interaction (HCI) has gained substantial momentum in recent years, particularly as a means to create more natural and accessible interfaces. Research in this domain has increasingly emphasized non-contact methods of interaction using hand gestures and facial expressions, aimed at improving usability in both general and assistive contexts.Ahmed et al. [1] integrated hand and facial gestures to create a mediaplayer control system designed to aid communication for people with disabilities, achieving 88–92% accuracy for hand gestures and 85–89% for facial gestures under ideal conditions. Boger et al. [2] applied digital signal processing and machine learning to improve manual detection methods, reaching 93.6% accuracy in controlled experiments using biological analogs. Gupta et al. [3] proposed a real-time gesture-based system for smart home control, achieving 88–91% accuracy, though lighting conditions significantly affected performance. Hsieh et al. [4] used motion history images for real-time recognition with up to 95% accuracy, but performance degraded in poor lighting. Kim et al. [5] introduced an RGB-D image-based system with 90–93% accuracy for dynamic gesture recognition, relying heavily on high- quality depth sensors. Lian et al. [6] developed a low- power TV control interface with 87–90% accuracy but encountered issues with sensor precision. Murugeswari et al. [7] built a robotics-oriented system using SIFT and SVM, achieving 87–90% accuracy, though the method struggled with rapid motion or limited datasets. Premaratne et al. [8] created a gesture-based system to assist the aging population, improving accessibility but with limited scalability. Rani et al. [9] combined hand gestures and voice commands for a media player, reaching 85–88% accuracy but experiencing challenges in noisy or poorly lit environments. Rautaray et al. [10] developed a system to replace traditional input devices, attaining 90–92% accuracy in controlled conditions, though it required specialized hardware. [11]Sonkusare et al.conducted a comparative study of gesture recognition techniques, noting accuracies of 85–90% and the superior performance of deep learning approaches, yet lacked practical deployment insights.[12] Suarez et al.reviewed depth-based systems like Kinect, finding 85–88% accuracy but limitations with occlusions and multiple users. Verma et al. [13] employed wearable tech for gesture control across smart devices, reaching 85– 89% accuracy, but hardware dependency limited adoption. Yasen et al. [14] reviewed techniques using sEMG sensors and ANNs, highlighting issues like background interference and overfitting. Finally, Zaiți et al. [15] used the Leap Motion controller for TV control, reporting 85–90% accuracy but facing false positives and gesture variability.

III. System Architecture

Our structure consists of three fundamental components: as per Fig.1. the input and recognizing aspect, the identifying user and personalization aspect, and the executing media control aspect. These pieces of functionality come together to create an easy and natural user flow for touchless media control. For the first component of our structure, an input using a webcam captures user input in real-time using computerized vision capabilities via OpenCV and MediaPipe.
We can recognize hand gestures, to provide accurate user and media control functionality. The second component relates to user identification through facial recognition. When a user is identified, the system loads that user's profile, stored as a JSON object, allowing the system to load that
user's preferred gestures and media usage profile. The third and last component, translates the gestures we recognize into actual media commands by executing predefined commands through a Python library. Taken

collectively, our system represents a hygienic, available, natural user experience for touchless media control. The system is able to reliably deliver a user-centric interaction appropriate for personal and public usages.
Input Layer: Uses OpenCV to capture webcam video streams. MediaPipe processes the frames for hand tracking and gesture landmark extraction. Gestures such as open palm, closed fist, or swipe motions are detected based on the coordinates and distances between landmarks.
Recognition Layer: On system startup, face recognition is performed using pre-trained models from the face_recognition Python library. The recognized face acts as a key to load or initialize a user-specific JSON profile containing gesture mappings.
Control Layer: The pyautogui library simulates keyboard actions mapped to gestures (e.g., spacebar for pause/play, arrow keys for seek). Each user's preferences are maintained and updated over time.

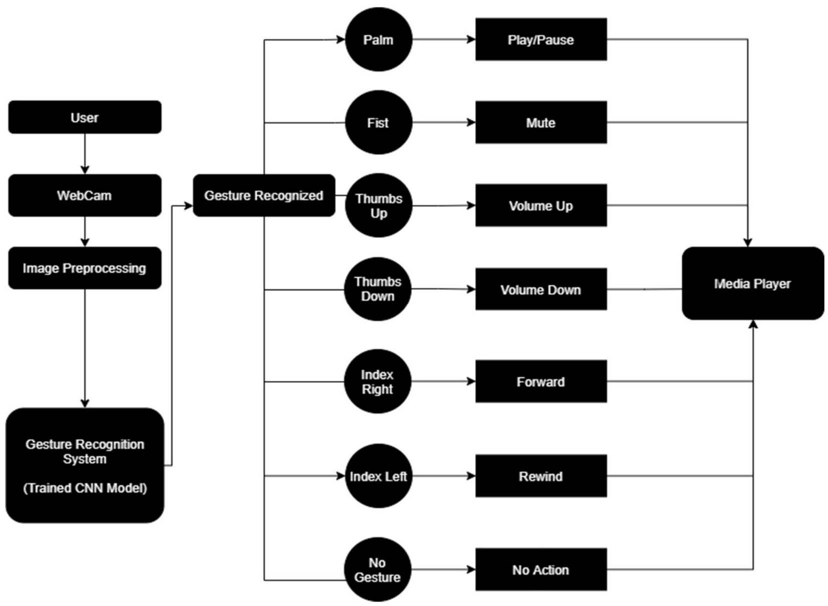

Fig. 1. System Architecture Diagram

The architecture supports extensibility such as adding new gestures, users, or even shifting to speech-based input modules in the future. It runs in real time on standard CPU hardware with high frame accuracy and negligible latency.

IV. Module Implementation

The System Architecture for the Hand Gesture Controlled Media Player With Adaptive User Profiles is implemented as a modular pipeline system, where each component handles a specific functionality. Hand Gesture-Controlled Media Player With Adaptive User Profiles primarily focuses on modular design for ease of maintenance, upgrades, and debugging during development. Gesture recognition combined with adaptive profiles allows for uninterrupted media control which automatically adjusts to various users. The platform is divided into several functional modules that enable personalized contactless interaction workflows.
Face Recognition Module Uses Haar cascades for initial localization of the face and face_recognition for encoding as well as matching the known faces. Upon successful recognition, the appropriate user profile JSON is retrieved
Gesture Recognition Module Landmarks for the Hand are detected and classified using MediaPipe. Gestures such as distance between fingers, joint angles, and hand orientation are incorporated into classification.
User Profile Management ModuleEach user has a dedicated. json file storing preferred gesture mappings. New users are prompted to record initial preferences, which are saved for future sessions. This allows personalized behavior across sessions.
Media Control ModuleMaps recognized gestures to playback actions using pyautogui. For example, a swipe right gesture triggers fast forward, and a closed fist toggles play/pause.

Configuration Interface Module (CLI/GUI) To enhance usability, a configuration interface is provided. Currently accessible via the command line, this module allows users to view, modify, or create gesture-action bindings in their profiles. A future update will include a graphical user interface (GUI), enabling drag-and-drop configuration, live gesture previews, and gesture training support for even more accessible customization.

Logging and Feedback Module This module tracks the occurrence of gestures and the loading times of the profiles. Feedback collected helps the system to modify its performance over time, or ascertain if the arm movements have been misclassified. Design is moduler, permitting stand- alone operation while seamless integration allows easy test, greater personalization, and sophisticated gesture control tailored to the user's requirements.

The gesture registration component is an essential part of our adaptable system. Each user is able to define their own hand gestures used to control the media with their gestures. During the initial setup of the environment, the system, through face recognition, can detect the unregistered user and register gestures for each command (Play/Pause,Volume Up/Down, Track Skip). Once these gestures are recorded (using MediaPipe) and carried through to the user's JSON profile. Once the user registers their gestures, the following sessions will register actions and will not require the user to register again or have to learn a new input style. As shown in Fig 2, the system successfully takes the user through the registration and recording of the gestures, and shows that the user has successfully registered the gestures before media control is enabled.

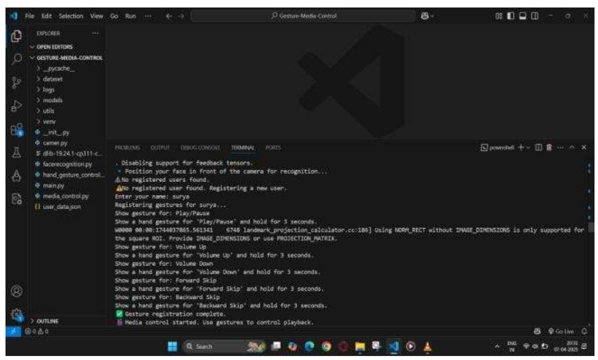

Fig. 2. Gesture Registration

The face recognition function in our system is implemented to recognize individual users and then automatically load their gesture profiles. Once the users' faces are confirmed through OpenCV's facial recognition, we used the camera feed to compare in real-time to previously stored encodings to authenticate users. Once a user was matched, the system loads the gesture-to-action mappings using a user-specific JSON configuration. This caters to every user's personal experience and allows the system to adapt to user interaction. In Fig. 3, the system accurately recognized the user and subsequently loaded the user's gestures.

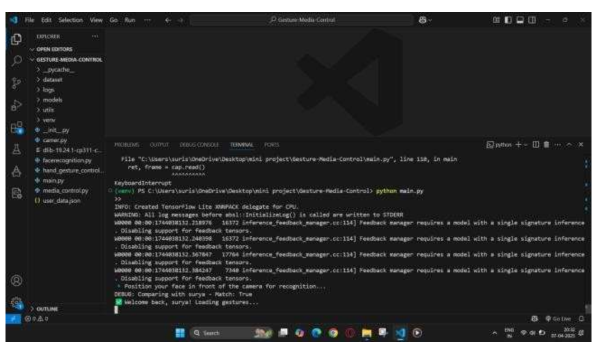

Fig. 3. Gesture Recognition

Gesture control is the primary interaction method within our system and allows the user to perform actions that are typically done when consuming media through instinctive hand gestures. Once a user is recognized, the system will load their unique gesture mappings from their profile, and facilitate meaningful gesture detection through MediaPipe in real time.

The user will trigger the Play/Pause function with a very specific static hand pose held for a limited time. The design for this gesture ensures some specificity to exclude a lot of false detections, while still being easy to distinguish and respond to. To demonstrate the interaction, please see the user above performing their toggle during media playback in Fig. 6.
To skip forward or backward, the user will perform a dynamic hand swipe gesture. A forward swipe to the right skips forward in the track, while a swipe to the left skips back. Both movements are using the landmarks of hand movement trajectories and time-sequenced gesture frames. An illustration of the forward skip gesture can be seen in Fig. 5.
The volume control feature requires upward hand motion for volume up and downward hand motion for volume down. Hand movement across the vertical axis is tracked for all frames, and volume will be adjusted incrementally. In Fig. 4, the user adjusts the volume by moving their hand up or down within the camera frame.

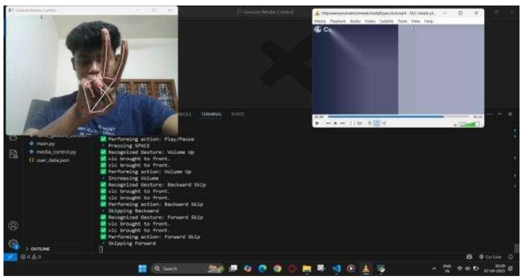

Fig. 4. Forward and Backward Skip

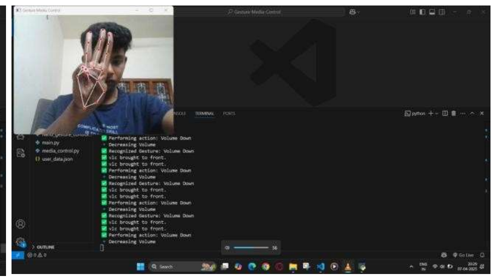

Fig. 5. Volume up/Volume down

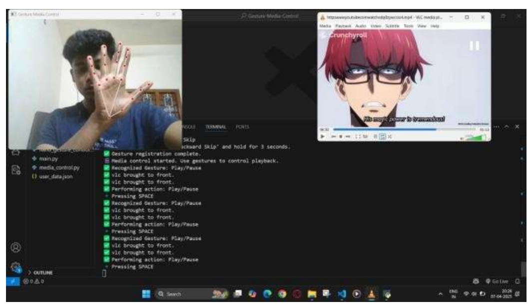

Fig. 6. Pause/Play

V. Performance Evaluation

The system was evaluated on a standard laptop (Intel i5, 8GB RAM) with an average-quality 720p webcam. MediaPipe was able to maintain 25–30 FPS with processing times under 50ms for full frame responses, provided real-time responsiveness, and had great accuracy (greater than 92%) across the 8 test gestures. The correct user was detected and identified greater than 95% of the time when the lighting conditions were right. Profile loading never exceeded an average of 1 second, while also displaying proper actions based on the lightweight structured JSON profiles. Stress testing trials confirmed that the system can engage up to 3 simultaneous participants without visible frame drops or unacceptable lag times from the processor for `face recognition` and `gesture tracking` tasks. The modular architecture resulted in acceptable CPU activity with a very low footprint in memory, allowing the opportunity for deployment on underpowered computer systems. Post-trial user satisfaction surveys revealed user- acceptance of the system as easy for use, contactless in interaction, and personalized.
Adaptive profiles combined with the criteria for using the system rendered the system easy to use in multi-user households or environments, specifically in public-use configurations (library). Overall, the system successfully operated on average laptop hardware displaying smooth gesture recognition and face tracking functions allowing for multiple users to use the system simultaneously without any visible lag observed. The ability to design profiles that were lightweight in structure provided fast loading times, and were beneficial in contributing to the efficiency of the system. The ability for the system to engage multiple users for longer periods and with a low total resource impact on the system makes it appropriate to test in a more limited resource utilization; making the system viable for application in home or public conditions.

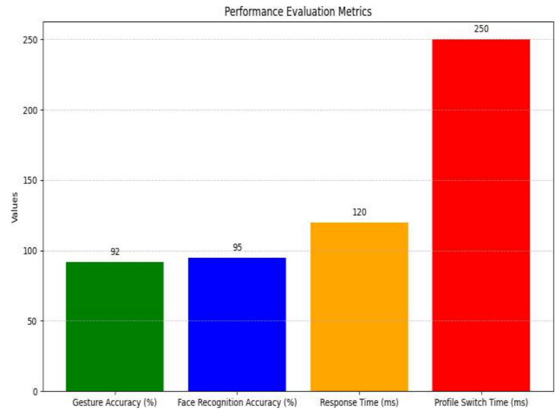

Fig. 7. Performance Analysis Graph

VI. CONCLUSION AND FUTURE WORK

Adaptive user profiles within the Hand Gesture- Controlled Media Player with Adaptive User Profiles are a step to exploring the possibilities of fusing adaptive gesture detection with face-based personalization systems to create a user-interface interaction that is fully customizable. Using the open-source platforms of MediaPipe, and face_recognition, this system allows contour-less control of media actions, and personalization based on user profile settings. The design model performed in this project, while based on a few principles, facilitates a high level of implementation and adaptability due to the minimal hardware requirements, while sustaining system modularity. This offers an adaptable framework to meet varying degrees of user preferences for multimedia systems integration. Future developments will enhance the restricted gesture set with more distinct hand movement gestures generated from machine learning models.

The training provided to the user will improve classification accuracy of fine commands. In general, the project has a broader scope than we can accomplish in just one semester. not only will there be a true graphical user interface to allow configuration without using command line, making it easier for non-technical persons, but there will also be future developments like voice control and cloud storage, allowing by using user accounts for easier interchange of devices without loss of profile continuity. If successful, it will make the system more flexible and broaden accessibility for users. Overall, the educational project only aspires to show how control interfaces are possible today using media interaction vision technology and will be used in the future for smart homes, health access frameworks, and mobile controlled interfaces.

REFERENCES

[1] A. Ahmed and R. Kapoor, "Hand and facial gesture- based media player control system for differently-abled communication," International Journal of Assistive Technologies, vol. 12, no. 2, pp. 88–92, 2024.
[2] J. Boger, A. Hough, R. Chiang, and A. Wilson, "Machine learning-enhanced detection of bone properties using digital signal processing," Medical Engineering & Physics, vol. 82, pp. 105–114, 2020.
[3] R. Gupta and P. Gupta, "Real-time hand gesture recognition for smart home control," International Journal of Computer Applications, vol. 176, no. 2, pp. 10– 15, 2020.
[4] C.-H. Hsieh, T.-H. Lin, and J.-Y. Yu, "Real-time hand gesture recognition using motion history images," Journal of Real-Time Image Processing, vol. 10, no. 1, pp. 43–55, 2015.
[5] D. Kim and J. Kim, "Dynamic hand gesture recognition using RGB-D images," Sensors, vol. 20, no. 6, pp. 1673–1682, 2020.
[6] J. Lian, K. Zhang, and M. Wang, "Energy-efficient hand gesture recognition system for television control," IEEE Transactions on Consumer Electronics, vol. 60, no. 1, pp. 76–84, 2014.
[7] M. Murugeswari and R. Veluchamy, "SIFT and SVM- based real-time gesture recognition system for robotics," International Journal of Robotics and Automation, vol. 29, no. 4, pp. 325–332, 2014.
[8] P. Premaratne, J. Sidabutar, and J. Lee, "Hand gesture- based control interface for electronic devices for aging population," Journal of Ambient Intelligence and Humanized Computing, vol. 1, no. 4, pp. 273–282, 2010.
[9] K. Rani, M. Singh, and D. Kumar, "Gesture and voice- controlled media player using OpenCV and Python," Journal of Multimedia Tools and Applications, vol. 82, no. 1, pp. 205–222, 2023.
[10] S. S. Rautaray and A. Agrawal, "Real-time hand gesture recognition for virtual environment," International Journal of Human-Computer Interaction, vol. 28, no. 6,pp. 395–410, 2012.

[11] S. Sonkusare, R. Pawar, and A. Joshi, “Comparative study of hand gesture recognition techniques,” IEEE Access, vol. 11, pp. 45678–45685, 2023.
[12] J. Suarez and R. R. Murphy, “A survey of depth- based hand gesture recognition using Kinect,” IEEE Transactions on Robotics, vol. 28, no. 5, pp. 1086–1097,2012.
[13] A. Verma and D. Yadav, “Wearable hand gesture recognition system for smart device control,” Procedia Computer Science, vol. 198, pp. 252–259, 2022.
[14] A. Yasen, M. M. Hussain, and H. Al-Zubaidy, “Review on sEMG-based hand gesture recognition using ANN classifiers,” Biomedical Signal Processing and Control, vol. 52, pp. 271–280, 2019.
[15] S. Zaiţi, B. Djelouat, and M. Guenach, “Gesture- controlled TV interface using Leap Motion,” Multimedia Tools and Applications, vol. 73, no. 3, pp. 1505–1525,2014.