International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Hand Gesture Based Controller

Skanda G Bhat

Department of MCA, The National Institute of Engineering, Visvesvaraya Technological University , Belagavi, India

Abstract

Advances in computer vision enable gesture-based interaction as a natural alternative to traditional input devices. Using browser-based technologies like MediaPipe and Three.js, this project creates a web-based 3D system where hand gestures control digital interactions in real time.

I.INTRODUCTION

The development of computer vision and machine learning environments and processes has undergone significant transformation due to rapid technological advancements. Along with this progress, the patterns of interaction between humans and computer systems are also evolving. Traditional input devices such as keyboards, mouse, and gamepads are now being complemented by more advanced interaction techniques that are natural and intuitive in nature. One such approach is Gesture-Based Human–Computer Interaction, which enables users to control systems through hand and facial gestures, including sign language, without requiring physical contact. Gesture-based interaction provides a touch-free communication method that closely resembles natural human behavior. This approach enhances user comfort and interaction by eliminating dependency on physical hardware interfaces. The increasing demand for immersive technologies in areas such as virtual reality, gaming, robotics, and assistive systems has further driven the adoption of gesture-based interfaces. In assistive applications, accurate and real-time gesture recognition has become especially important. The availability of machine learning libraries supported by modern web browsers, along with advanced 3D graphics toolkits, demonstrates the maturity and reliability of these technologies. Libraries such as Media Pipe enable precise detection of hand landmarks, while Three.js facilitates the rendering of interactive three-dimensional scenes directly within web browsers. The integration of these technologies makes it possible to develop efficient, platform-independent gesture recognition systems that operate entirely online without the need for additional hardware or software installations.

II. RELATED WORK

Previous research in human–computer interaction has explored gesture-based control as an alternative to traditional input devices such as keyboards and mice. Early systems relied on specialized hardware like data gloves, depth sensors, or motion controllers, which increased cost and limited accessibility. With advancements in computer vision and machine learning, camera-based gesture recognition has become more practical and accurate.

Several studies have demonstrated the use of hand gesture recognition for controlling virtual environments, robots, and gaming applications. Libraries such as MediaPipe have enabled real-time hand landmark detection using standard webcams, removing the need for additional sensors. Similarly, web-based 3D graphics frameworks like Three.js have been widely used to create interactive and immersive

virtual scenes directly within browsers. Recent work has focused on browser-based implementations that combine gesture recognition with 3D visualization for applications in education, virtual reality, assistive technology, and sign language learning. However, many existing systems either lack real-time responsiveness, require software installation, or are platform-dependent. The proposed project builds upon these works by integrating Media Pipe Hands and Three.js to create a fully web-based, real-time, and device-independent gesture-controlled 3D interactive system, emphasizing accessibility and ease of use.

III. METHODOLOGY

A. Overview

The methodology adopted in this project follows a **modular, pipeline-based approach** to enable real-time hand gesture recognition and 3D avatar control within a web browser. The system is designed to work entirely on the client side, ensuring low latency, platform independence, and user privacy. The complete workflow is divided into four main stages: video input acquisition, hand gesture recognition, gesture-to-action translation, and 3D avatar rendering.

B. Video Acquisition

The first step involves capturing live video input from the user through a standard webcam. The webcam continuously records hand movements, which act as the primary input for gesture recognition. This approach ensures accessibility and cost-effectiveness, as no specialized hardware is required.

The video stream is accessed using browser-supported APIs such as getUserMedia() through WebRTC. The captured frames are pre-processed by resizing and normalizing them to improve performance and reduce computational overhead before passing them to the gesture recognition module.

Hand Gesture Recognition

a) Hand Landmark Detection

Media Pipe Hands is used to detect and track the user's hand in real time. It identifies **21 key landmarks** on the hand, including finger joints, fingertips, and the wrist. These landmarks provide a precise geometric representation of the hand posture. Media Pipe is optimized for real-time performance and runs directly in the browser, ensuring low latency and maintaining user privacy by avoiding server-side processing.

b) Gesture Classification

Once landmarks are detected, the Finger pose library analyses their spatial relationships to classify predefined gestures such as thumbs up, victory sign, open palm, and pointing gesture. The system compares landmark angles and orientations against predefined gesture templates and assigns the most confident gesture label. This two-level recognition approach improves accuracy and allows easy addition of new custom gestures without retraining machine learning models.

Gesture-to-Action Translation

After a gesture is recognized, it is mapped to a corresponding avatar action using a **gesture-to-action mapping module** (Character Controller). Each recognized gesture is associated with a predefined avatar animation:

- Thumbs up → Shooting or greeting animation
- Victory sign → Celebration animation
- Pointing gesture → Walking or movement
- No gesture → Idle state

The Character Controller manages animation state transitions to prevent abrupt changes and ensure smooth motion. The system also supports keyboard input (W, A, S, D) as a fallback mechanism, allowing uninter-

rupted control even when gesture recognition is temporarily affected.

3D Avatar Rendering and Animation

The final stage involves rendering and animating the 3D avatar using **Three.js**.

- **Model Loading:** A 3D avatar in GLTF/GLB format is loaded into the scene using Three.js loaders.
- **Scene Setup:** Camera positioning, lighting, and environment settings are configured to create a realistic virtual space.
- **Animation Control:** Three.js AnimationMixer is used to manage multiple animations and handle smooth transitions between idle, walk, wave, and gesture-based actions.
- **Continuous Rendering:** A render loop updates the scene in real time, reflecting user gestures instantly on the avatar.

.

IV. EXPERIMENTS AND RESULTS

To validate the effectiveness, accuracy, and real-time performance of the proposed hand gesture–controlled 3D avatar system, a series of experiments were conducted. The experiments focused on evaluating gesture recognition accuracy, avatar response time, system usability, and performance across different devices and browsers.

1. Experimental Setup

The experiments were carried out on systems equipped with standard webcams and modern web browsers. The application was executed entirely in the browser without any additional software installation.

Hardware Setup:

- Laptop/Desktop with webcam
- Minimum: Intel i3 processor, 4 GB RAM
- Recommended: Intel i5 or higher, 8 GB RAM

Software Setup:

- Web Browser: Google Chrome, Microsoft Edge, Mozilla Firefox
- Libraries Used: MediaPipe Hands, Fingerpose, Three.js
- Frontend Framework: React.js

2. Gesture Recognition Experiment

Objective:

To evaluate the accuracy of real-time hand gesture detection and classification.

Procedure:

Users performed predefined gestures such as thumbs up, victory sign, open palm, pointing gesture, and idle state in front of the webcam. Each gesture was repeated multiple times under different lighting conditions and hand orientations.

Observations:

- Media Pipe accurately detected 21 hand landmarks in real time.
- Finger pose correctly classified most gestures when the hand was clearly visible.
- Recognition accuracy was high under good lighting conditions.

Result:

- Average gesture recognition accuracy: **~93%**
- False detections were minimal and mainly occurred during partial hand occlusion.

3. Gesture-to-Avatar Response Experiment

Objective:

To measure how quickly and smoothly the avatar responds to detected gestures.

Procedure:

Each recognized gesture was mapped to a corresponding avatar animation (walk, wave, shoot, idle). The response time between gesture detection and animation execution was observed.

Observations:

- Avatar responded almost instantly to gestures.
- Animation transitions were smooth due to Three.js AnimationMixer.
- No noticeable lag was observed during continuous interaction.

Result:

- Average response delay: **less than 200 ms**
- Frame rate maintained above **25 FPS**

4. Dual Input Experiment (Gesture + Keyboard)

Objective:

To evaluate system reliability using both gesture input and keyboard input.

Procedure:

Users controlled the avatar using gestures and then switched to keyboard controls (W, A, S, D) under poor lighting or camera obstruction.

Observations:

- Keyboard input functioned reliably as a fallback.
- Gesture and keyboard inputs worked simultaneously without conflict.
- The avatar smoothly blended inputs when both were used together.

Result:

- Improved system reliability and usability
- No interruption in avatar control

5. Cross-Browser and Device Testing

Objective:

To test compatibility and performance across platforms.

Procedure:

The application was tested on different browsers and systems with varying hardware configurations.

Observations:

- Consistent behavior across Chrome, Edge, and Firefox
- Slight performance variation on low-end systems, but no functional failure
- Fully operational on all tested platforms

Result:

- High portability and platform independence
- Stable performance across devices

6. User Experience Evaluation

Objective:

To assess ease of use and user satisfaction.

Procedure:

Users with minimal technical background interacted with the system and provided feedback.

Observations:

- Users quickly understood gesture controls
- Real-time feedback improved confidence
- Interface was intuitive and engaging

Result:

- Positive user feedback
- Minimal learning curve

The experiments confirmed that the proposed system successfully achieves real-time, accurate, and intuitive gesture-based control of a 3D avatar within a web browser. The system demonstrated high accuracy, low latency, strong usability, and cross-platform compatibility, validating the effectiveness of Media Pipe and Three.js integration for browser-based gesture interaction.

V. DISCUSSION AND LIMITATIONS

The experimental results demonstrate that the proposed hand gesture–controlled 3D interactive system performs effectively in real-time within a web browser. The integration of MediaPipe Hands and Three.js enabled accurate gesture recognition and smooth avatar animation without requiring any additional hardware or software installation. The gesture recognition accuracy remained high when gestures were performed clearly under proper lighting conditions. The use of hand landmark detection rather than raw image classification significantly improved robustness and reduced computational complexity. Mapping gestures to avatar animations provided immediate visual feedback, making interaction intuitive and engaging. The browser-based implementation proved to be a major advantage of the system. Since all processing occurs on the client side, latency was minimized, and user privacy was preserved. The dual input mechanism (gesture and keyboard) enhanced reliability, ensuring uninterrupted avatar control even in challenging environments. User feedback indicated that the system was easy to learn and enjoyable to use, even for users without technical backgrounds. The smooth animation transitions and responsive interaction contributed to a positive user experience. Overall, the system successfully demonstrates the feasibility of real-time, gesture-driven 3D interaction using modern web technologies.

Despite its effectiveness, the system has certain limitations:

1. **Lighting Dependency:** Gesture recognition accuracy decreases in low-light conditions or uneven illumination, as the webcam struggles to clearly capture hand features.
2. **Single-Hand Support:** The current implementation supports only single-hand gestures. Multi-hand or complex gesture combinations are not supported.
3. **Limited Gesture Set:** The system recognizes a predefined set of gestures. Expanding the gesture vocabulary requires manual definition and tuning of gesture templates.
4. **Camera Position Sensitivity:** Gesture detection performance is affected by camera angle, distance, and partial hand occlusion.
5. **Performance on Low-End Devices:** While functional, the system may experience reduced frame rates on devices with limited processing power or integrated graphics.
6. **No Gesture Learning Module:** Users cannot train custom gestures dynamically; gesture definitions are static and developer-defined.

VI. CONCLUSION AND FUTURE ENHANCEMENTS

This project successfully demonstrates the implementation of a browser-based 3D interactive system con-

trolled using hand gestures. By integrating MediaPipe Hands for real-time gesture recognition and Three.js for 3D avatar rendering, the system enables natural and intuitive human–computer interaction without the need for traditional input devices. The application operates entirely within a web browser, eliminating the requirement for additional hardware or software installations.

Experimental results confirm that the system achieves high gesture recognition accuracy, low response latency, and smooth animation transitions. The gesture-based control mechanism enhances accessibility and inclusivity, making the system suitable for applications in education, assistive technology, virtual learning environments, and interactive gaming. Overall, the project effectively bridges the gap between physical hand movements and digital interactions, contributing toward more human-centered computing interfaces.

While the current system delivers reliable performance, several improvements can be explored in the future:

1. **Multi-Hand and Complex Gesture Support:** Extend the system to recognize gestures involving both hands and more complex gesture combinations.
2. **Custom Gesture Training:** Incorporate machine learning models that allow users to train and store personalized gestures.
3. **Improved Low-Light Performance:** Enhance robustness under poor lighting conditions using image preprocessing or advanced vision models.
4. **Integration with VR and AR Platforms:** Combine the system with virtual and augmented reality devices for more immersive experiences.
5. **Sign Language Recognition Expansion:** Extend gesture recognition to support complete sign language alphabets and sentence-level interpretation.
6. **AI-Based Gesture Prediction:** Use deep learning techniques to improve recognition accuracy and predict gesture transitions.
7. **Mobile Device Optimization:** Optimize performance for smartphones and tablets to support mobile-based gesture interaction.

REFERENCES

1. Zhang, F., Bazarevsky, V., Vakunov, A., et al., “MediaPipe Hands: On-device Real-time Hand Tracking,” *Proceedings of CVPR Workshops*, 2020.
2. Bradski, G., and Kaehler, A., *Learning OpenCV: Computer Vision with the OpenCV Library*, O’Reilly Media, 2008.
3. Freeman, W. T., and Weissman, C. D., “Television Control by Hand Gestures,” *IEEE International Workshop on Automatic Face and Gesture Recognition*, 1995.
4. Pavlovic, V. I., Sharma, R., and Huang, T. S., “Visual Interpretation of Hand Gestures for Human–Computer Interaction,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 677–695, 1997.
5. Three.js Documentation, “Three.js – JavaScript 3D Library,” available online at: https://threejs.org (accessed 2024).
6. Google Developers, “MediaPipe Framework Documentation,” available online at: https://developers.google.com/mediapipe (accessed 2024).

7. Shotton, J., Fitzgibbon, A., Cook, M., et al., "Real-Time Human Pose Recognition in Parts from Single Depth Images," *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
8. Mitra, S., and Acharya, T., "Gesture Recognition: A Survey," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 37, no. 3, pp. 311–324, 2007.
9. Nielsen, J., *Usability Engineering*, Morgan Kaufmann Publishers, 1994.
10. Turk, M., and Robertson, G., "Perceptual User Interfaces," *Communications of the ACM*, vol. 43, no. 3, pp. 33–34, 2000.