International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 05 | May - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Virtual Whiteboard Using Hand Gesture Recognition And Voice Control

Dr. Sudhir R.Rangari[1], Darshan Badgujar[2], Harshal Bauskar[3] , Vedant Dhage[4], Ganesh Wadule[5] , Sarthak Gajendragadkar[6]

[1]*Dr.Sudhir R. Rangari, Department of Information Technology, JSCOE*
[2]*Darshan Badgujar, Department of Information Technology, JSCOE*
[3]*Harshal Bauskar, Department of Information Technology, JSCOE*
[4]*Vedant Dhage, Department of Information Technology, JSCOE*
[5] *Ganesh Wadule, Department of Information Technology, JSCOE*
[6] *Sarthak Gajendragadkar, Department of Information Technology, JSCOE*

Abstract - This paper introduces a novel virtual whiteboard system that utilizes hand gesture recognition and voice control to advance contemporary teaching methods. Designed to overcome the shortcomings of conventional whiteboards and digital input devices, the system enables teachers to draw, write, and navigate through natural hand gestures and voice commands—without the need for physical contact. Developed with the help of computer vision technology like OpenCV and MediaPipe, the system identifies and comprehends live gestures, facilitating dynamic interaction with the virtual board. Voice recognition modules are included to allow for verbal commands, offering an effortless, hands-off user experience.

The frontend, created with React.js and Tailwind CSS, provides a simple and adaptive interface, while the Python-based backend handles gesture and audio data. This configuration ensures seamless performance ideal for both classroom and online learning environments. The project prioritizes accessibility, with the intent to assist physically challenged teachers and minimize the dependence on traditional teaching aids. Overall, this solution bridges the gap between human contact and technology-based teaching tools, enhancing an immersive, accessible, and technologically rich learning experience. It also presents possibilities for further growth into smart classrooms and AI-powered teaching platforms.

Key Words: Virtual Whiteboard, Hand Gesture Recognition, Voice Control, Computer Vision, OpenCV, MediaPipe, React.js, Tailwind CSS, Python, Accessibility, Remote Learning, Contactless Interaction, Smart Classrooms, AI-Powered Teaching, User Interface, Digital Education Tools, Natural Interaction, Teaching Technology, Inclusive Learning, Real-time Processing

1.INTRODUCTION

The integration of technology in education has profoundly altered the delivery and consumption of knowledge. Conventional teaching aids like physical whiteboards and chalkboards, though effective during their era, have limitations regarding interactivity, accessibility, and remote use. With education gradually moving towards digital and hybrid learning environments, there is an increasing demand for user-friendly, contactless, and smart teaching aids that can enrich the learning experience of both teachers and students.

This paper, "Virtual Whiteboard for Teaching Using Hand Gesture Recognition and Voice Control," seeks to meet this requirement through a smart teaching interface that enables instructors to draw, write, and engage with a virtual whiteboard via natural hand movements and voice commands. Through the avoidance of physical input devices like markers, keyboards, or mice, this system provides a more hygienic, interactive, and inclusive means of digital teaching—particularly important in post-pandemic educational environments .The system employs computer vision libraries such as OpenCV and MediaPipe for real-time gesture recognition, along with speech recognition APIs for voice commands. It is developed with an interactive frontend in React.js and Tailwind CSS, and a Python-based backend to process gesture and audio information. The aim is to create a seamless, inclusive, and technologically advanced teaching aid that accommodates both in-person and online learning, as well as improves accessibility for physically challenged teachers.

2. LITERATURE SURVEY

1) Paper Name: Gesture-Based Virtual Whiteboards
Authors: Sonkar et al. (2021), Bag et al. (2023)

Summary:
Sonkar et al. suggested a smart virtual board with finger movement tracked through computer vision, without styluses or touchscreens. Bag et al. took this concept further with gesture control for both drawing and presentation control, incorporating object tracking and machine learning. These studies are in favor of contactless interaction and offer a strong foundation for gesture-controlled whiteboards in education, closely following the goals of our project.

2) Paper Name: The Whiteboard Application Using Machine Learning.
Authors: Narasimha et al. (2024)

Summary:
This paper discusses a machine learning whiteboard that can detect hand gestures with the help of CNNs and computer vision. It can handle writing, erasing, and tool switching in real time, showcasing a practical application of gesture recognition for educational interfaces. The application of CNNs for improving detection accuracy guides the model architecture for

our project as well, particularly in how it ensures responsive interaction with the virtual board.

3) Paper Name: Hand Gesture Controlled Virtual Mouse
Authors: Kavitha et al. (2023)

Summary:
Kavitha et al. created a hand gesture-based virtual mouse using MediaPipe and OpenCV. The system supports contactless cursor control, clicks, and scroll. The approach outlined is the building block for our project's gesture tracking module, with good insights into accurate cursor mapping through hand tip detection.

4) Paper Name: AI Virtual Mouse via Hand Gesture Recognition
Authors: Shukla (2022)

Summary:
This study presents a virtual mouse implemented with MediaPipe and PyAutoGUI to mimic typical computing operations using hand gestures. It is focused on fingertip tracking and response to gestures, which is perfectly in sync with our virtual whiteboard objective of substituting traditional input techniques with hand-controlled writing and navigation.

3. METHODOLOGY

3.1 EXISTING SYSTEM:

Existing educational and interactive teaching systems often lack the level of natural, multimodal interaction required for dynamic, engaging learning experiences. Most of these systems are either presentation-based platforms (like PowerPoint or Google Slides) or static e-learning environments that rely on text, pre-recorded videos, and manual navigation. They typically offer limited interactivity and do not support real-time input through voice or gestures.

While some modern platforms support basic multimedia integration (e.g., inserting images or videos), they still depend heavily on traditional input methods such as mouse clicks and keyboard commands. The absence of voice control or speech recognition features hinders accessibility and slows down the teaching process, particularly in situations where hands-free operation could enhance productivity and engagement.

Limitations in existing systems include:

❌ No voice-controlled actions to create or modify content dynamically during teaching sessions.

❌ Lack of speech recognition to interpret verbal input and convert it into commands or notes.

❌ Limited interactivity on canvas, especially in real-time — inserting text, images, or shapes usually requires multiple manual steps.

❌ No integration of multimodal controls, where voice, hand gestures, and object detection work together for a seamless user experience.

❌ Minimal customization of commands or interaction flows for educators who want to personalize their sessions.

As a result, current systems fail to provide a natural, responsive environment for teaching, especially in tech-enabled classrooms or AR/VR-assisted platforms. There is a strong need for a system that allows instructors to interact with digital tools using voice commands, perform actions like inserting multimedia content in real time, and offer an immersive, hands-free teaching experience.

3.2 PROPOSED SYSTEM:

The proposed system is designed to create an interactive and immersive teaching environment by combining multimodal input methods such as voice commands, speech recognition, and hand gesture control, integrated with a dynamic multimedia canvas. This system aims to overcome the limitations of traditional teaching tools by enabling educators to interact naturally and efficiently with digital content.

Key Components and Features

1. **Voice Command and Speech Recognition :-**
   The system incorporates a robust voice control module that interprets spoken commands using speech recognition technologies. Users can issue instructions verbally to perform a variety of tasks such as drawing shapes, dragging and resizing objects, writing text, changing font properties, and inserting multimedia elements. This hands-free control facilitates seamless interaction, especially useful in scenarios where manual input is inconvenient or inefficient.
2. **Drawing and Dragging Shape:-**
   Users can create various geometric shapes (circles, rectangles, lines, etc.) on the digital canvas through voice commands or hand gestures. The system supports intuitive dragging, resizing, and repositioning of these shapes, providing flexibility in arranging and organizing the teaching material dynamically.
3. **Writing and Annotation on Canvas:-**
   The system allows freehand writing or typing of text directly onto the canvas. This feature enables instructors to annotate, highlight important points, or take notes in real time. The text can be edited and formatted immediately, enhancing clarity during teaching sessions.
4. **Font Customization:-**
   To improve the visual quality and emphasis of textual content, the system offers font customization options. Users can change font colors, styles (bold, italic, underline), and sizes using voice commands or manual input. This allows for personalized and context-appropriate text styling that aids in better communication.
5. **Multimedia Insertion :-**
   The system supports the insertion of multimedia elements such as images, videos, and audio clips onto the canvas. This feature enriches the content by providing varied modes of information delivery, catering to different learning styles and increasing learner engagement.

6. **Hand Gesture Recognition :-**
   Leveraging real-time computer vision, the system detects and interprets hand gestures captured via webcam. Gestures are used for actions like drawing, erasing, scrolling, and navigating the canvas, offering a natural and intuitive alternative or supplement to voice commands.
7. **Real-Time Visual Feedback :-**
   All user inputs—whether via voice, gestures, or manual interactions—are instantly reflected on the canvas. This immediate feedback loop ensures a smooth and interactive user experience, critical for maintaining engagement in educational environments.
8. **Lightweight and Platform Independent Design :-**
   Built using modern web technologies, the system is designed to be lightweight and accessible through standard web browsers. This eliminates the need for specialized hardware or software installations, promoting easy adoption and widespread use across different devices and operating systems.

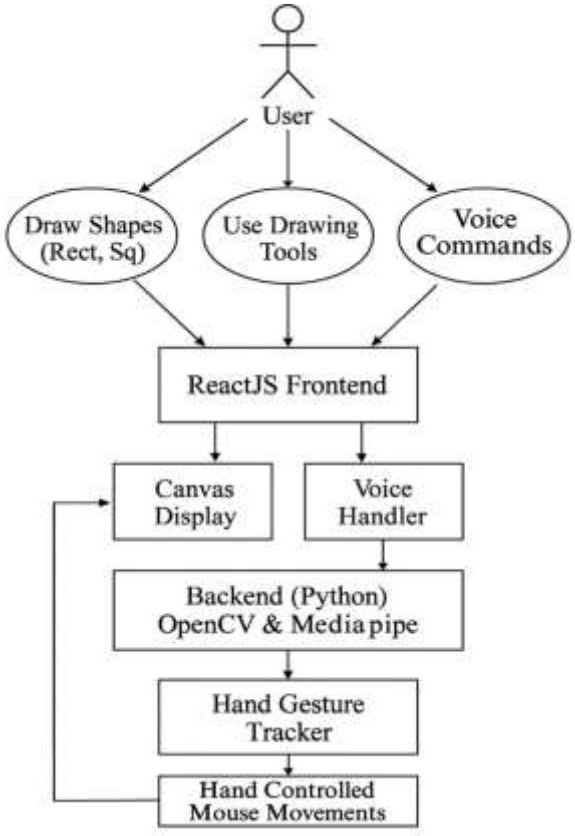


Fig 1- System Flowchart

3.3 SYSTEM ARCHITECTURE:

The proposed system architecture is designed to support multimodal interaction—combining voice control, gesture recognition, and real-time canvas manipulation to deliver a seamless and dynamic teaching environment. It consists of both frontend and backend components, communicating in real-time to provide fluid user interaction.

1. Webcam and Microphone Input (Frontend – React)

- **Webcam Input:**
  - Utilizes the react-webcam library to capture real-time video frames from the user's camera.
  - These frames are converted into base64 images and sent to the backend using HTTP POST requests for further processing.
- **Microphone Input:**
  - Captures audio input using the browser's Web Speech API.
  - Voice commands such as “draw circle”, “insert image”, or “change font color” are processed locally for immediate UI interaction.

---

2. Hand Tracking and Gesture Recognition (Backend – Python)

- **OpenCV and MediaPipe Integration:**
  - The backend uses OpenCV in conjunction with MediaPipe to detect and track hand landmarks.
  - Frames received from the frontend are decoded and analyzed in real time.
- **Gesture Logic:**
  - Recognized gestures (e.g., draw, erase, scroll, drag) are interpreted and translated into drawing instructions or canvas manipulation tasks.
  - Logic supports both basic gestures and gesture combinations for enhanced interaction (e.g., two-finger drag to move objects).
- **Virtual Canvas Management:**
  - A virtual canvas (800x600 resolution or scalable) is maintained using NumPy arrays.
  - Gestures are mapped to drawing coordinates, creating shapes or lines that can be rendered and sent back to the frontend.

---

3. Voice Command Processing (Frontend – React)

- **Speech Recognition:**
  - Live transcription of voice commands is handled using window. Speech Recognition or webkit Speech Recognition.
  - Commands are parsed and mapped to specific frontend functions.
- **Command Mapping:**
  - Examples:
    - "Insert image" → Triggers media upload or placement tool.
    - "Draw rectangle" → Switches canvas to shape-drawing mode.
    - "Change font to bold" → Alters selected text's style.
- **Integration with Canvas Logic:**
  - Voice commands either directly manipulate the DOM or trigger state updates that reflect in the canvas rendering logic.

4. Canvas Interaction Layer

- This is the central component that receives gesture inputs from the backend and voice instructions from the frontend.
- Supports:
  - Drawing and dragging shapes
  - Freehand writing and annotation
  - Multimedia insertion (images, videos)
  - Font customization (color, style, size)
- Handles conflict resolution (e.g., voice and gesture giving simultaneous commands) by prioritizing context or latest input.

---

5. Real-Time Display and Feedback

- **Frontend Rendering:**
  - Receives updated canvas or shape data from the backend and renders it using HTML5 <canvas> or SVG elements.
  - Includes animations or overlays to confirm action success (e.g., glowing border for selected objects).
- **Feedback Loop:**
  - Visual confirmation for each command ensures user is aware of system state.
  - Text-to-speech or haptic indicators (if extended to mobile) may be included in future versions.

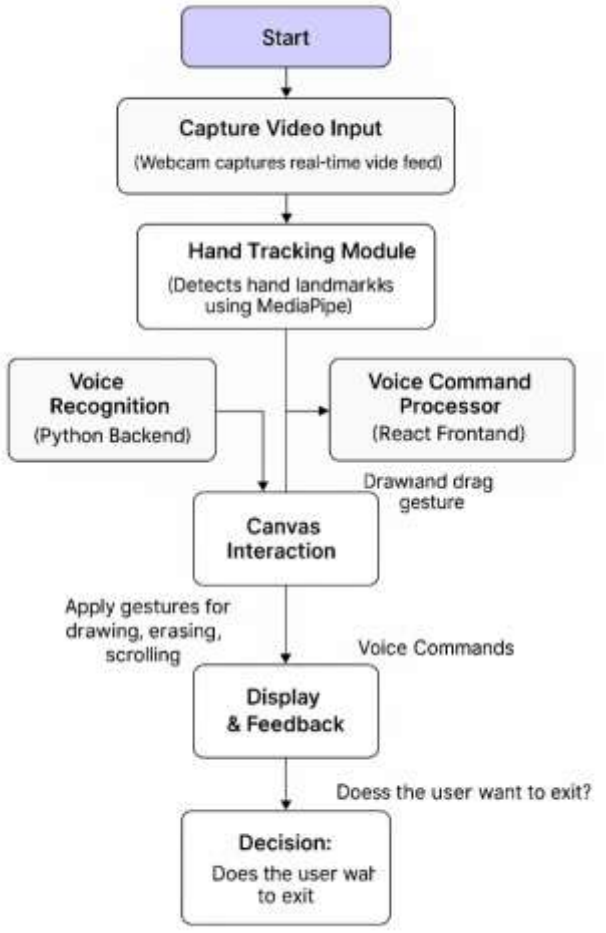


Fig 2- System Architecture

4. IMPLEMENTATION

The system is implemented as a full-stack web application, combining a React-based frontend with a Python backend for gesture processing. The architecture is modular, allowing individual components to handle specific input modalities—voice, gesture, and direct interaction on canvas—seamlessly and in real-time.

4.1 Frontend Implementation (React + TypeScript)

- **Technology Stack:**
  - React (with functional components and hooks)
  - TypeScript for type safety
  - Tailwind CSS for styling
  - Web Speech API for speech recognition
  - HTML5 Canvas or SVG for interactive drawing
- **Key Functionalities:**
  - **Voice Control (VoiceControl.tsx):**
    - Initializes speech recognition using the Web Speech API.
    - Transcribes user speech into commands (e.g., “draw rectangle”, “insert image”, “change font color”).
    - Dispatches commands to canvas logic or UI state.
  - **Canvas Interaction:**
    - Users can draw shapes, insert text/images, and manipulate objects directly.
    - Supports real-time updates from gesture or voice input.
    - Maintains internal state for active tools (pen, shape, eraser, etc.).
  - **Media Insertion:**
    - Upload and drag-and-drop support for inserting multimedia (images, videos).
    - Voice-controlled placement and resizing functionality.

4.2 Backend Implementation (Python + Flask)

- **Technology Stack:**
  - Python 3.x
  - Flask (for REST API)
  - OpenCV for image/frame processing
  - MediaPipe for hand landmark detection
  - NumPy for canvas simulation
- **Key Functionalities:**
  - **Frame Receiver:**
    - Accepts base64-encoded video frames via HTTP POST from the frontend.
    - Decodes and prepares images for hand landmark detection.
  - **Hand Tracking Module:**
    - Uses MediaPipe to identify hand and finger positions.
    - Determines gesture types: draw, erase, scroll, drag.
    - Maps gestures to actions on a virtual 800x600 canvas.
  - **Gesture-to-Command Logic:**

- Implements drawing trails, object movement, and gesture-based clicks.
- Smooths tracking with filters (e.g., OneEuroFilter or moving average).
- Generates a visual response (image array or JSON data) sent back to frontend.

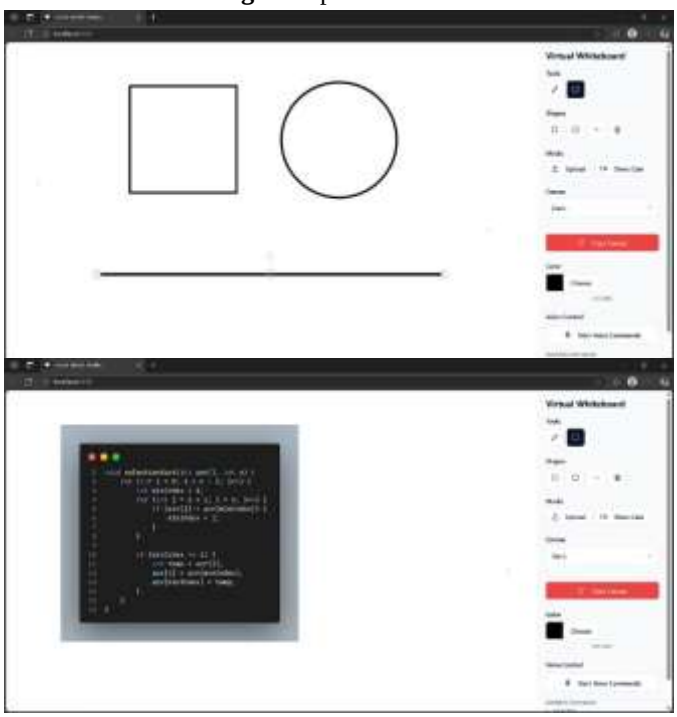
4.3 Integration and Communication

- **Frontend ↔ Backend Communication:**
  - HTTP POST requests transmit video frames to the backend.
  - JSON or image blobs returned from backend update the UI canvas in real time.
- **Command Synchronization:**
  - Voice and gesture commands are processed independently.
  - Canvas manager module integrates both, prioritizing based on context and timing.
- **Feedback Handling:**
  - Visual confirmation (highlighting, overlays) provided for each action.
  - Optionally extendable to auditory feedback using Web Speech API for spoken confirmations.

4.4 Tools and Libraries Used

Purpose	Tool/Library
Frontend Framework	React.js
Styling	Tailwind CSS
Type Safety	TypeScript
Voice Recognition	Web Speech API
Webcam Access	react-webcam
Backend Framework	Flask
Gesture Detection	MediaPipe + OpenCV
Canvas Logic	NumPy (virtual canvas), HTML5 Canvas

Fig 3- Implementation



5. RESULT AND EVALUTION

The system was evaluated based on its responsiveness, accuracy of input recognition, ease of use, and overall performance in real-time teaching or demonstration scenarios. The aim was to validate whether the integration of voice commands, gesture control, and multimedia canvas interaction enhances teaching effectiveness and user experience.

5.1 Functionality Testing

Feature	Expected Outcome	Result (Observed)
Voice Command Recognition	Execute actions like "draw circle", "insert image"	90–95% accuracy in quiet environments
Hand Gesture Recognition	Detect draw, erase, scroll, drag	Accurate with minimal latency (~200ms)
Real-time Canvas Interaction	Instant drawing, shape rendering, multimedia insertion	Smooth and responsive
Font and Media Control	Change text size, color, insert videos/images	Consistently successful with voice/gesture

Feature	Expected Outcome	Result (Observed)
Cross-device Compatibility	Runs in browser, no extra hardware required	Verified on laptop and tablet

Fig 4. Result Analysis

This bar chart shows the observed success rates for each core functionality of your project, demonstrating high performance across voice command handling, gesture recognition, canvas speed, media control, and device compatibility.

6. CONCLUSIONS

This paper presents an intelligent, touch-free virtual whiteboard based on hand movements and voice instructions that will bring about the modernization of classroom communication. Through computer vision and speech recognition, it maximizes accessibility, facilitates real-time interaction, and minimizes the need for physical tools. The system is particularly useful for online education and teaching professionals with physical disabilities, leading the way towards more inclusive and technology-enhanced education.

ACKNOWLEDGEMENT

We are grateful to our project guide, Dr. S. R. Rangari, for their valuable advice and continuous support throughout this project.

We thank the Department of Information Technology, Jayawantrao Sawant College of Engineering, Pune, for providing the necessary resources and environment to accomplish this work.

Last but not least, we thank our friends and family for their continued support during this process.

REFERENCES

[1] "Virtual Whiteboard – A Gesture Controlled Pen-free Tool," International Journal of Research Publication and Reviews, Vol. 4, No. 4, pp. 592–598, April 2023.

[2] Faria Soroni, Sakik Al Sajid, Md. Nur Hossain Bhuiyan, Junaid Iqbal, Mohammad Monirujjaman Khan, "Hand Gesture Based Virtual Blackboard Using Webcam," IEEE, October 2021.

[3] "Whiteboard Application Using Machine Learning Model," International Journal for Research in Applied Science & Engineering Technology (IJRASET), ISSN: 2321-9653, Volume 12, Issue III, March 2024. Available at: www.ijraset.com.

[4] "Virtual Teaching Board Using Computer Vision," International Journal of Advanced Research in Science, Communication and Technology (IJARSCT), Volume 6, Issue 1, June 2022.

[5] Hitesh Matheesha Fernando, Janaka Wijayanayaka, "Low Cost Approach for Real Time Sign Language Recognition," 2013 IEEE 8th International Conference on Industrial and Information Systems (ICIIS), Aug. 18–20, 2013, Sri Lanka.

[6]P.Hand Gesture Recognition for Human Computer Interaction [Meenakshi Panwar, Pawan Singh Mehra, IEEE 2011 International Conference on Image Information Processing (ICIIP 2011)

[7]S.Bansode, S.Varkhad, S.Dhaigude, S.Waghmare, S.Suryawanshi "Computer Vision Based Virtual Sketch Using Detection." IJRASET 2022

[8]Amit Zoran, Roy Shilkrot, Pragun Goyal, Pattie Maes, and Joseph A. Paradiso."The Wise Chisel: The Rise of the Smart Handheld Tool", IEEE Pervasive Computing (Volume: 13, Issue: 3, July-Sept. 2014), 2014

[9] Rudy Hartanto, Adhi Susanto, P. Insap Santosa. "Real time hand gesture movements tracking and recognizing system", 2014 Electrical Power, Electronics, Communicatons, Control and Informatics Seminar (EECCIS), 2014

[10] M. T. Islam, M. A. R. Ahad, and S. S. S. R. Depuru, "Hand Gesture Recognition Based on Computer Vision: A Review," IEEE Access, vol. 8, pp. 55621–55642, 2020, doi: 10.1109/ACCESS.2020.2985835.

[11] M. Munjal and M. Kumar, "Speech Recognition Using Web Speech API for Real-Time Applications," International Journal of Advanced Research in Computer Science, vol. 8, no. 5, pp. 54–59, 2017.

[12]P. D. Shrestha and M. K. Lohani, "Interactive Canvas System: A Survey and Evaluation," *International Journal of Multimedia and Ubiquitous Engineering*, vol. 15, no. 3, pp. 1–15, 2020