School of Software Engineering

Hand Gesture Controlled Presentation

Group Members:
Osman Muhudin Ali (2120246007)
Gebreslasie Berhane Zere (2120246059)
Medhanie Yonatan Haile (2120246018)

Course: Web Development

June 21, 2025

Contents

1 Introduction and Project Overview

The Hand Gesture Controlled Presentation System is an innovative application that enables users to control PowerPoint presentations using hand gestures captured through a webcam. This technology eliminates the need for traditional input devices such as keyboards, mice, or remote controls, providing a more natural and intuitive presentation experience. The system leverages computer vision techniques to detect hand positions and gestures, translating them into commands that control the presentation flow and allow for real-time annotation capabilities.

This project addresses the growing need for contactless interaction with digital content, particularly relevant in educational, corporate, and conference settings. By implementing a gesture-based control system, presenters can maintain their position and focus on audience engagement rather than being tethered to a device for slide navigation.

2 Functionality and Features

The system provides the following key functionalities:

1. **PowerPoint Integration**:
   - Automatic extraction of slides from PowerPoint presentations (.pptx/.ppt files)
   - Conversion of slides to image format for manipulation
2. **Gesture-Based Navigation**:
   - Forward navigation (next slide) using pinky finger gesture
   - Backward navigation (previous slide) using thumb gesture
   - Gesture threshold adjustment to accommodate different user heights and positions
3. **Real-Time Annotation**:
   - Drawing capabilities using index finger
   - Pointer function using index and middle fingers
   - Annotation erasure using three fingers (index, middle, ring)
4. **User Interface**:
   - Intuitive GUI for presentation selection and configuration
   - Live webcam feed with hand tracking visualization
   - Transparent gesture threshold line for user guidance
   - Picture-in-picture display showing the user's webcam feed alongside presentation content

3 Implementation Approach

3.1 System Architecture

The project follows a modular architecture with two main components:

1. **GUI Module**: Handles user interaction, PowerPoint processing, and application initialization
2. **Hand Tracking and Presentation Control Module**: Manages webcam input, hand detection, gesture recognition, and presentation manipulation

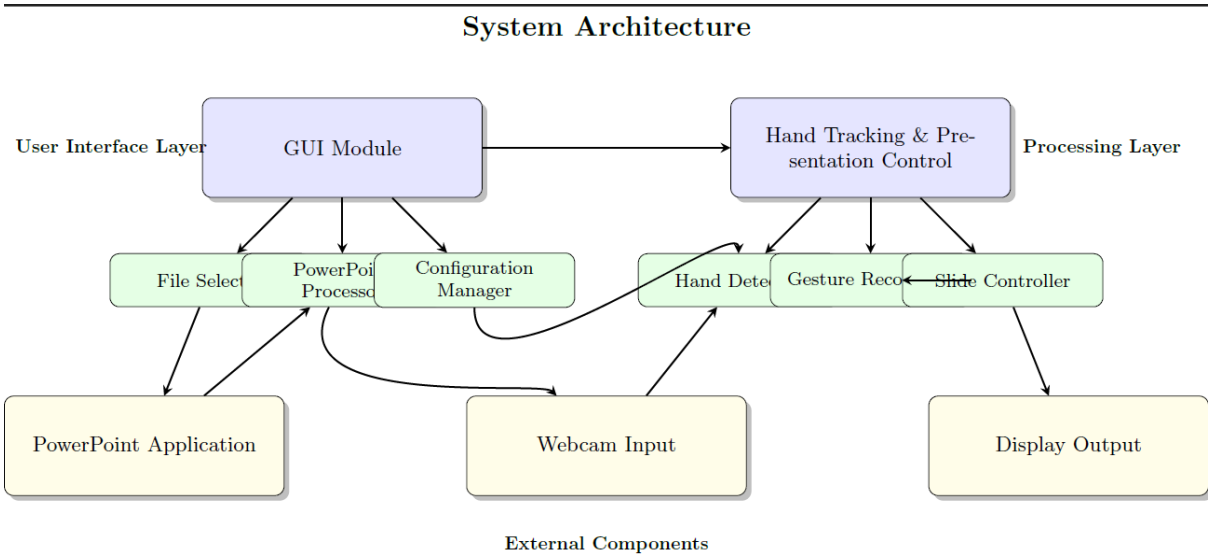

Figure 1: System architecture diagram showing the main modules, submodules, and their interactions

The system workflow follows these steps:

1. User selects a PowerPoint file through the GUI
2. System extracts slides as images to a temporary folder
3. Webcam activates and begins hand tracking
4. User gestures above the threshold line are interpreted as commands
5. Presentation is navigated and annotated according to detected gestures
6. Temporary files are cleaned up upon application exit

3.2 Technical Implementation

PowerPoint Processing:

- COM interface utilized to interact with PowerPoint application
- Slides exported as high-resolution PNG images (1280×720)

- Sequential naming ensures proper slide order
- Unique temporary folder creation to avoid conflicts

Hand Detection and Tracking:

- MediaPipe and OpenCV libraries for hand landmark detection
- 21-point hand skeleton model to identify finger positions
- Center point calculation for threshold detection
- Finger up/down state detection using relative landmark positions

Gesture Recognition Logic:

- Thumb up only [1,0,0,0,0] → Previous slide
- Pinky up only [0,0,0,0,1] → Next slide
- Index and middle up [0,1,1,0,0] → Pointer mode
- Index up only [0,1,0,0,0] → Drawing mode
- Index, middle, and ring up [0,1,1,1,0] → Erase last annotation

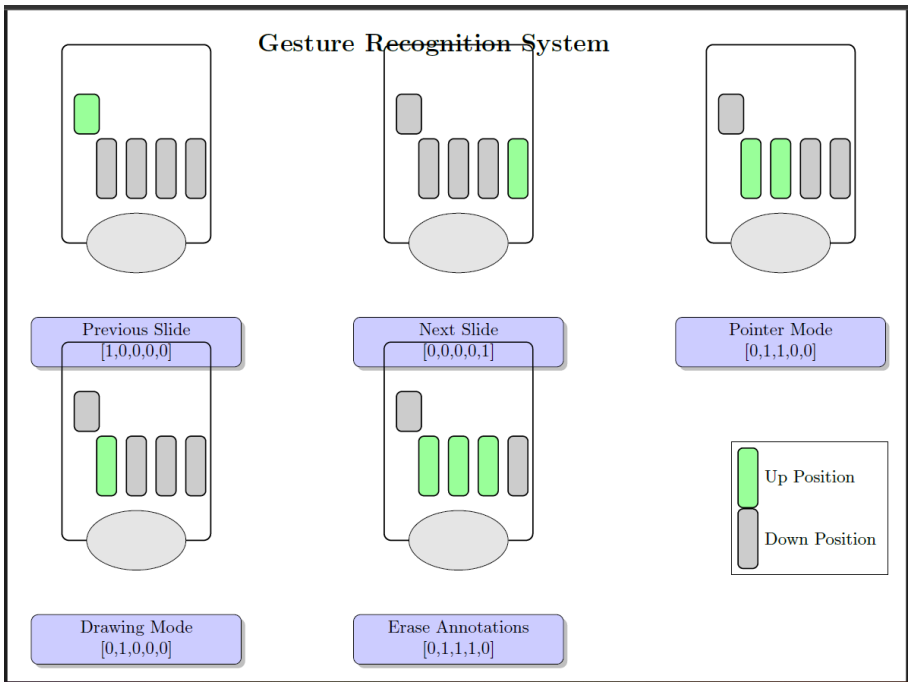

Figure 2: Hand gesture recognition system showing finger configurations and corresponding actions

Annotation System:

- Vector-based storage of drawing points
- Line rendering between sequential points
- Annotation persistence across slides
- Per-slide annotation collection

4 Technology Stack

4.1 Core Technologies

- **Programming Language**: Python 3.x
- **Computer Vision**: OpenCV (4.7.0.72) - Image processing and webcam interaction
- **Hand Tracking**: MediaPipe (0.10.11) - Hand landmark detection framework
- **Hand Gesture Module**: CVZone (1.5.6) - Simplified hand tracking implementation
- **Numerical Processing**: NumPy (1.23.5) - Array manipulation for coordinates
- **GUI Framework**: Tkinter - Native Python interface for application window
- **PowerPoint Integration**: COM automation via comtypes - Programmatic control of PowerPoint

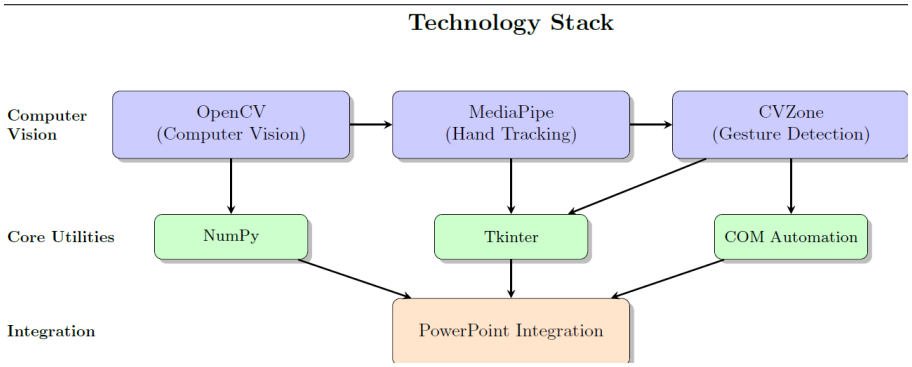

Figure 3: Technology stack hierarchy showing dependencies and relationships between core technologies

4.2 Dependencies

The project relies on the following libraries:

```
cvzone==1.5.6
opencv-python==4.7.0.72
numpy==1.23.5
mediapipe==0.10.11
python-pptx (for PowerPoint file parsing)
comtypes (for PowerPoint COM interaction)
```

5 Development Environment

5.1 Hardware Requirements

- Computer with webcam (minimum 720p resolution recommended)

- Processor: Minimum dual-core 2.0 GHz (quad-core recommended)
- RAM: Minimum 4GB (8GB recommended)
- Display resolution: 1280×720 or higher
- Sufficient lighting for hand detection

5.2 Software Requirements

- Windows 10/11 operating system
- Microsoft PowerPoint installed (for slide extraction)
- Python 3.7 or higher
- Required Python packages (via requirements.txt)
- Virtual environment (recommended)

5.3 Development Tools

- Python IDE (VS Code, PyCharm, etc.)
- Git for version control
- PowerPoint for test presentation creation

6 User Interaction and Experience

6.1 User Interface

The application provides two main interfaces.

1. **Configuration Interface**:
    - File selection through standard dialog
    - Gesture threshold adjustment slider
    - Start and quit buttons
    - Status information display
2. **Presentation Interface**:
    - Main presentation slide display
    - Webcam feed picture-in-picture
    - Visual gesture threshold line
    - Real-time hand tracking visualization
    - Annotation display

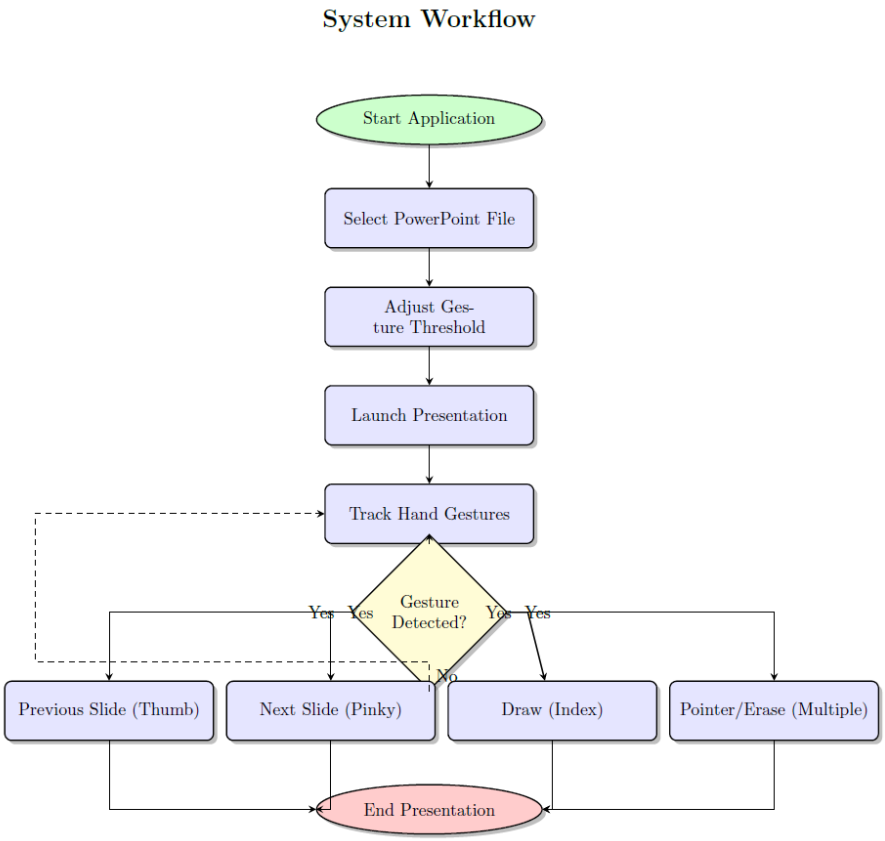

Figure 4: Enter CaptionSystem workflow diagram showing user interaction flow and command execution process

6.2 Interaction Flow

1. User launches the application via GUI.py
2. User selects a PowerPoint presentation file
3. User adjusts the gesture threshold if needed
4. User clicks "Start Presentation" to begin
5. Webcam activates and starts tracking hand movements
6. User navigates slides and annotates using defined gestures
7. User can exit by pressing 'q' key or closing the window

6.3 Gesture Sensitivity and Accessibility

The application includes a customizable gesture threshold to accommodate:

- Different user heights
- Various webcam positioning
- Different arm lengths
- Accessibility needs

The threshold appears as a horizontal green line across the webcam feed, indicating the point above which gestures are recognized.

7 Testing and Validation

7.1 Testing Methodology

The application was tested using the following approaches:

1. **Component Testing**:
    - Hand detection accuracy in various lighting conditions
    - PowerPoint extraction reliability with different file formats
    - Gesture recognition precision with different users
2. **Integration Testing**:
    - End-to-end workflow from file selection to presentation control
    - Error handling during PowerPoint processing
    - Resource management (file cleanup)
3. **User Experience Testing**:
    - Gesture learning curve for new users
    - Comfort during extended use
    - Annotation precision

7.2 Performance Considerations

- Real-time processing requires efficient implementation
- Frame rate maintained above 20 FPS for responsive control
- Memory usage monitored for slide extraction of large presentations
- Error handling prevents crashes due to webcam issues or file problems

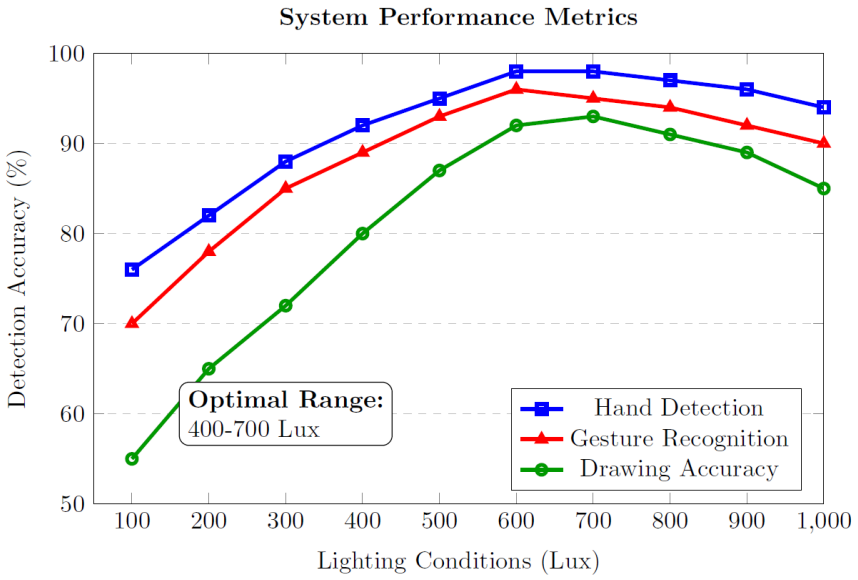

Figure 5: Performance metrics showing system accuracy and responsiveness under different lighting conditions

8 Limitations and Future Improvements

8.1 Current Limitations

1. **Dependence on Microsoft PowerPoint**:
   - Requires PowerPoint installation for COM automation
   - Limited to Windows operating system
2. **Environmental Factors**:
   - Lighting conditions affect hand detection accuracy
   - Background complexity can impact tracking
   - Physical space needed for gesture performance
3. **Technical Constraints**:
   - Single-hand tracking only
   - Limited gesture vocabulary
   - No persistent annotations between sessions

8.2 Potential Enhancements

1. **Cross-platform Compatibility**:
   - Implementation for macOS and Linux
   - Support for alternative presentation formats (PDF, Google Slides)
2. **Advanced Interaction**:
   - Two-hand gesture support
   - Voice command integration
   - Custom gesture programming
   - Zoom and pan capabilities
3. **Feature Expansion**:
   - Annotation saving and loading
   - Recording capabilities
   - Virtual laser pointer with custom colors
   - Gesture-based text input

9 Deployment and Usage

9.1 Installation Instructions

1. Ensure Python 3.7+ is installed
2. Clone or download the project repository
3. Install dependencies:
   ```
   pip install -r requirements.txt
   ```
4. Ensure Microsoft PowerPoint is installed
5. Launch the application:
   ```
   python GUI.py
   ```

9.2 Usage Guidelines

1. **Preparation**:
   - Position webcam at appropriate height
   - Ensure adequate lighting
   - Clear background for better detection
   - Prepare PowerPoint presentation
2. **Execution**:

- Launch application and select presentation
- Adjust threshold to match your position
- Start presentation and use gestures above the green line
- Use thumb for previous slide, pinky for next slide
- Use index finger to draw, index+middle to point
- Use three fingers to erase annotations

3. **Troubleshooting**:
   - If hand detection is unreliable, adjust lighting or background
   - If PowerPoint extraction fails, check file format compatibility
   - Restart application if webcam fails to initialize

10 Conclusion

The Hand Gesture Controlled Presentation System demonstrates the potential of computer vision and gesture recognition in creating more natural human-computer interactions. By eliminating the need for physical input devices, the system enables presenters to focus on content delivery while maintaining a dynamic and engaging presentation style.

The modular architecture and integration of modern computer vision libraries provide a robust foundation for future expansion and enhancement. While current limitations exist regarding environmental dependencies and platform constraints, the core functionality successfully delivers an intuitive and responsive presentation control system.

This project serves as both a practical tool for enhancing presentations and as a demonstration of how gesture recognition can transform traditional interaction paradigms. The techniques employed can be extended to other domains requiring touchless control systems, from medical applications to public information kiosks.

11 References

1. MediaPipe Hand Tracking: https://google.github.io/mediapipe/solutions/hands.html
2. OpenCV Documentation: https://docs.opencv.org/
3. CVZone Library: https://github.com/cvzone/cvzone
4. Python COM Automation: https://pythoncom.sourceforge.net/
5. Tkinter Documentation: https://docs.python.org/3/library/tkinter.html