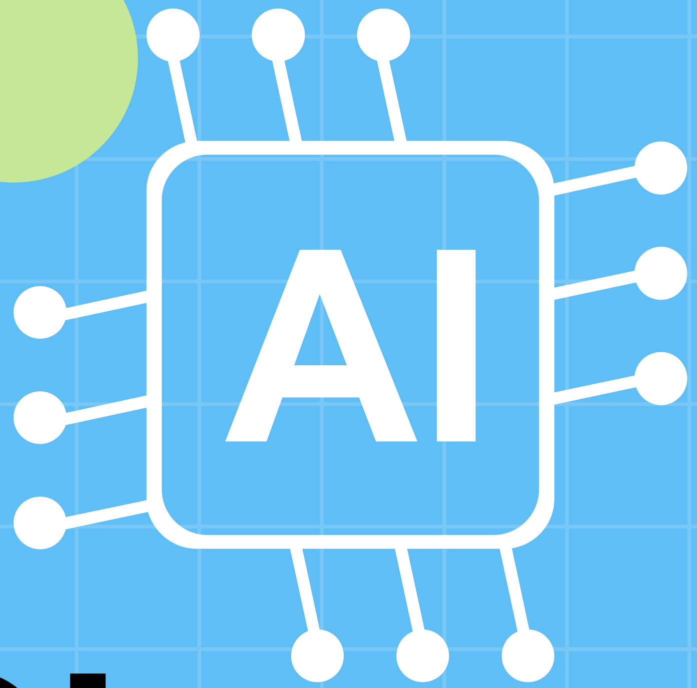
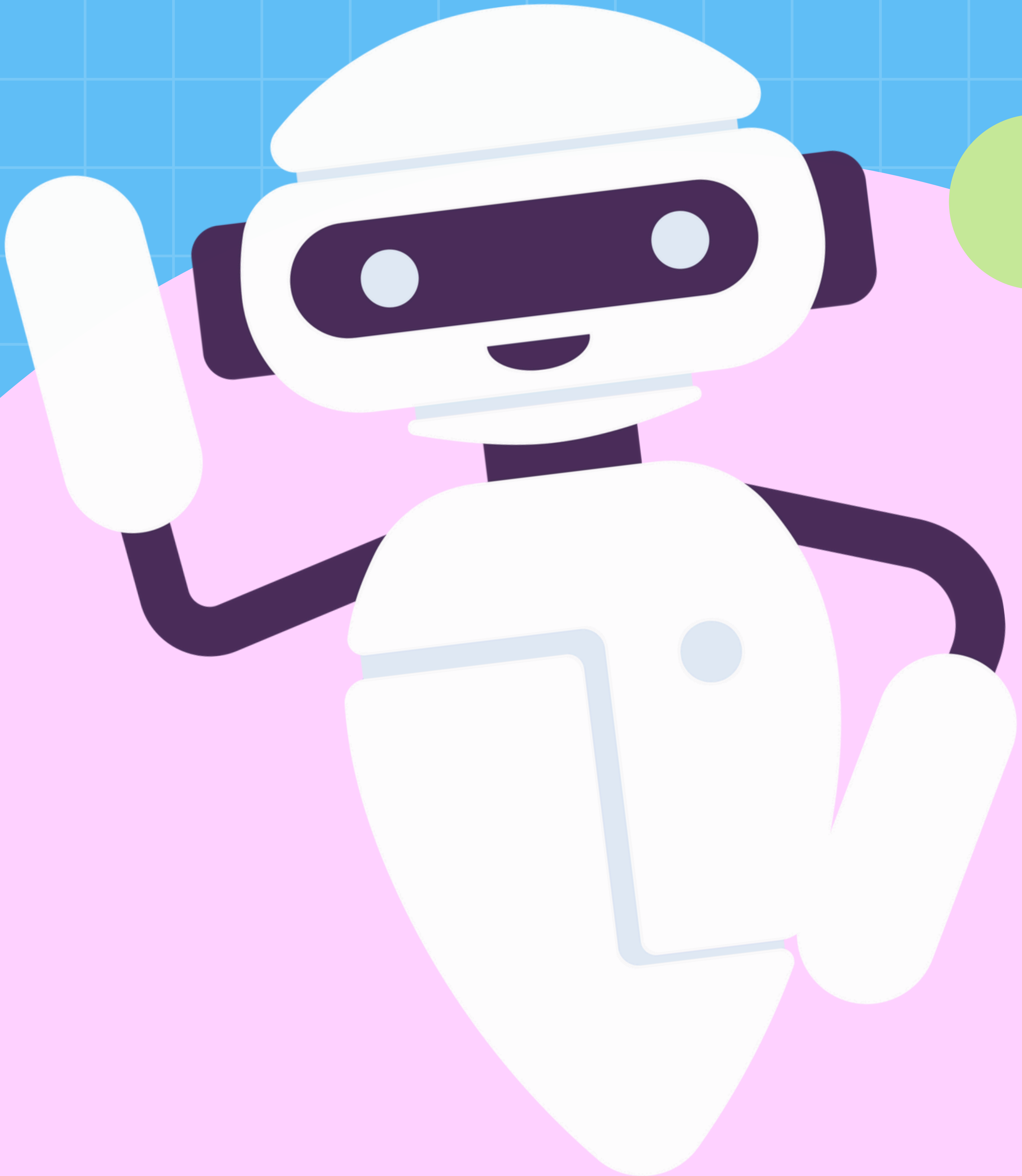


GESTUREVOL

Control your sound with a wave of your hand

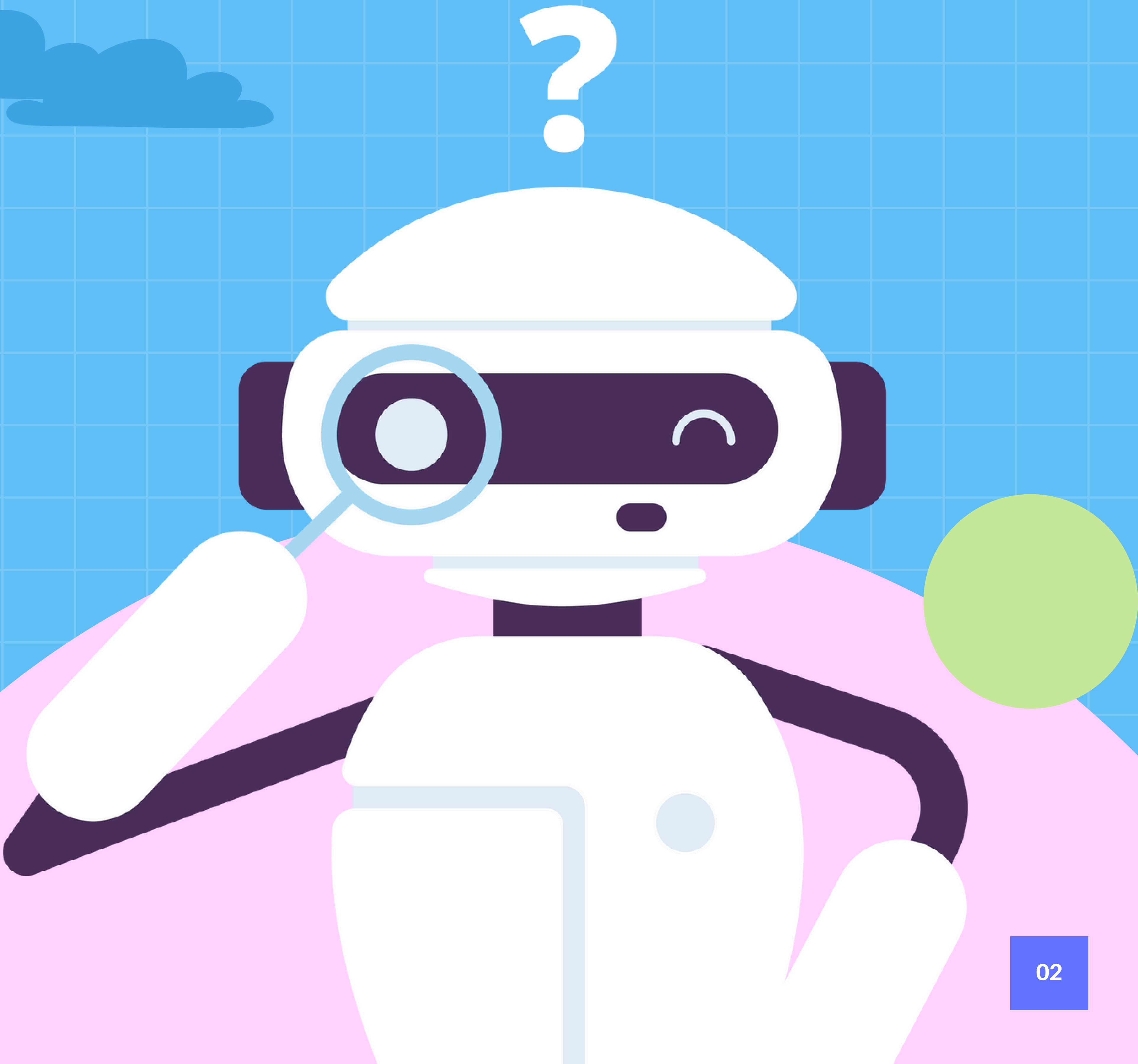


KRISH JHAWAR	RA2311033010070
RISHIKA SARKAR	RA2311033010071
SECTION	AJ2

INTRODUCTION

GestureVol is a vision-based system for touchless volume control using hand gestures.

- It uses **MediaPipe and OpenCV** for accurate, real-time gesture recognition.
- Achieves **98.2% accuracy,** ensuring reliable and responsive performance.
- Addresses limitations of existing systems such as **lighting sensitivity, limited gesture vocabulary, and high latency.**
- Enables **hygienic, accessible control** without the need for specialized hardware.
- Works with a **standard webcam, making it cost-effective and easy to deploy.**
- Supports intuitive gestures like swipe, pinch, and finger counts for media control.



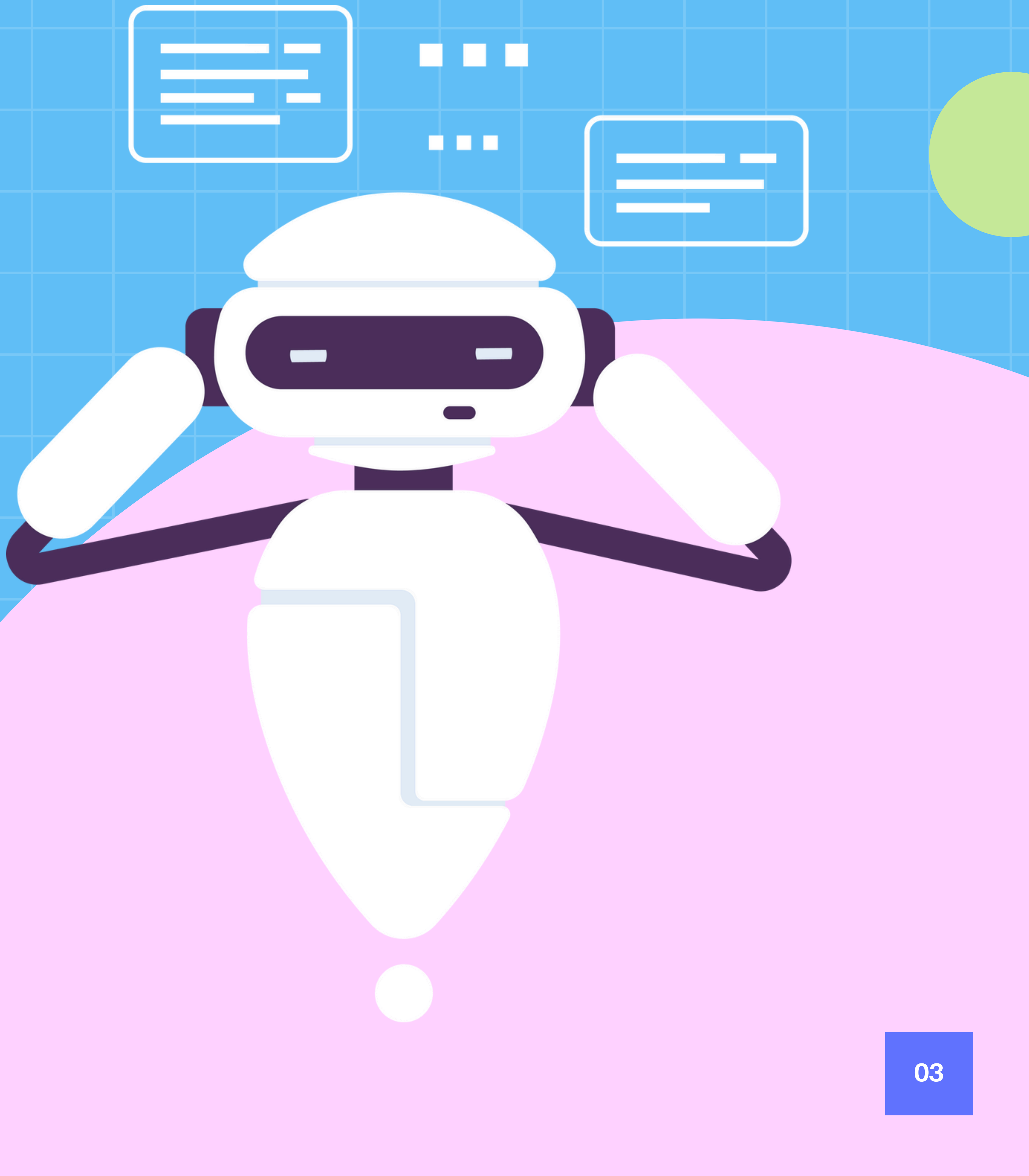
PROBLEM STATEMENT

Physical volume controls pose hygiene and accessibility issues. Existing gesture systems struggle with:

- Poor lighting adaptability
- Limited gestures
- High lag
- No user feedback

These flaws make them impractical for daily use. Our system solves this by:

- **Adapting to lighting changes**
- **Supporting multiple gestures**
- **Running smoothly on standard hardware**
- **Providing clear visual feedback**
- **Delivers reliable, real-world touchless interaction.**



LITERARY SURVERY 1

TITLE	HAND GESTURE RECOGNITION AND VOLUME CONTROL
YEAR	2022
FINDINGS	Uses Haar cascades in OpenCV for real-time hand detection. Qualitative analysis of volume adjustment.
ADVANTAGES	1. Simple implementation 2. No hardware dependencies 3. Works with basic webcams
DISADVANTAGES	1. Low accuracy (~85%) 2. Sensitive to lighting 3. Limited to static gestures

LITERARY SURVERY 2

TITLE	AUTOMATED HAND GESTURE SOFTWARE FOR VOLUME CONTROL
YEAR	2023
FINDINGS	MediaPipe tracks finger angles; achieves 99.3% accuracy at 1.5s latency.
ADVANTAGES	1. Extremely high accuracy 2. Real-time processing 3. No markers/gloves needed
DISADVANTAGES	1. Requires calibration 2. Limited to 5 gestures 3. CPU-intensive

LITERARY SURVERY 3

TITLE	VOLUME CONTROLLER USING HAND GESTURES
YEAR	2023
FINDINGS	OpenCV+NumPy system with 95% success in varied lighting.
ADVANTAGES	1. Robust to lighting changes 2. No special hardware 3. Intuitive swipe gestures
DISADVANTAGES	1. Only volume control 2. 5% error rate 3. Requires clear hand visibility

LITERARY SURVERY 4

TITLE	REAL-TIME ML GESTURE RECOGNITION
YEAR	2024
FINDINGS	MediaPipe+Pycaw achieves 45ms latency with dynamic scaling.
ADVANTAGES	1. Near-instant response 2. Precise volume increments 3. Handles multiple users
DISADVANTAGES	1. Complex setup 2. Requires GPU for best performance 3. High power consumption

LITERARY SURVERY 5

TITLE	INNOVATIVE TECHNIQUES FOR REAL-TIME CONTROL
YEAR	2023
FINDINGS	Haar+CNN hybrid with 96.3% accuracy in labs.
ADVANTAGES	1. High precision 2. No depth sensors needed 3. Works with OpenCV
DISADVANTAGES	1. Fails in cluttered backgrounds 2. Needs retraining for new gestures 3. High RAM usage

LITERARY SURVERY 6

TITLE	RASPBERRY PI VOLUME CONTROL
YEAR	2022
FINDINGS	RPi+OpenCV works best at 640x480 (100% at 0.5m).
ADVANTAGES	1. Ultra-low cost ($35) 2. Portable 3. Energy efficient
DISADVANTAGES	1. Accuracy drops after 1m 2. Low FPS (15-20) 3. No multi-hand support

LITERARY SURVERY 7

TITLE	BRIGHTNESS+VOLUME CONTROL
YEAR	2024
FINDINGS	Controls 3 parameters via MediaPipe gestures.
ADVANTAGES	1. Multi-functional 2. User-friendly UI 3. Open-source
DISADVANTAGES	1. Conflicting gestures 2. Requires steady hands 3. No haptic feedback

LITERARY SURVERY 8

TITLE	AR/VR VOLUME CONTROL
YEAR	2023
FINDINGS	MediaPipe for VR environments with palm tracking.
ADVANTAGES	1. Immersive AR/VR integration 2. Sub-100ms latency 3. Markerless
DISADVANTAGES	1. VR headset required 2. High GPU demands 3. Limited to slow gestures

RESEARCH GAP

POOR LOW-LIGHT PERFORMANCE

Adaptive exposure control + histogram equalization

Code: cap.set(cv2.CAP_PROP_EXPOSURE, -4) + adaptive_brightness_contrast()

NO GESTURE VARIETY

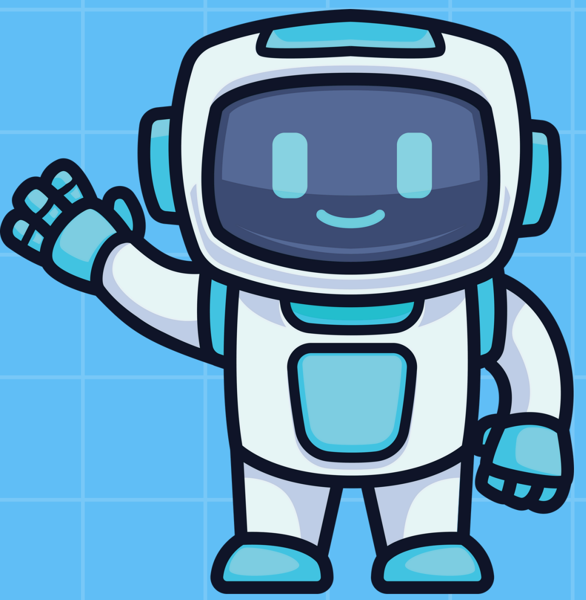
Multi-command detection (mute/play/volume)

Code: detect_gesture() with finger state analysis

MANUAL CALIBRATION

Auto-scaling based on hand size

Code: area = (bbox[2] - bbox[0]) * (bbox[3] - bbox[1])



ACCIDENTAL TRIGGERS

Gesture hold-time threshold (2s)

Code:if time.time() - last_gesture_time > GESTURE_HOLD_TIME

PLATFORM DEPENDENCY

Cross-platform audio control (Windows/Linux)

Code: PLATFORM = platform.system() + Pycaw/ALSA

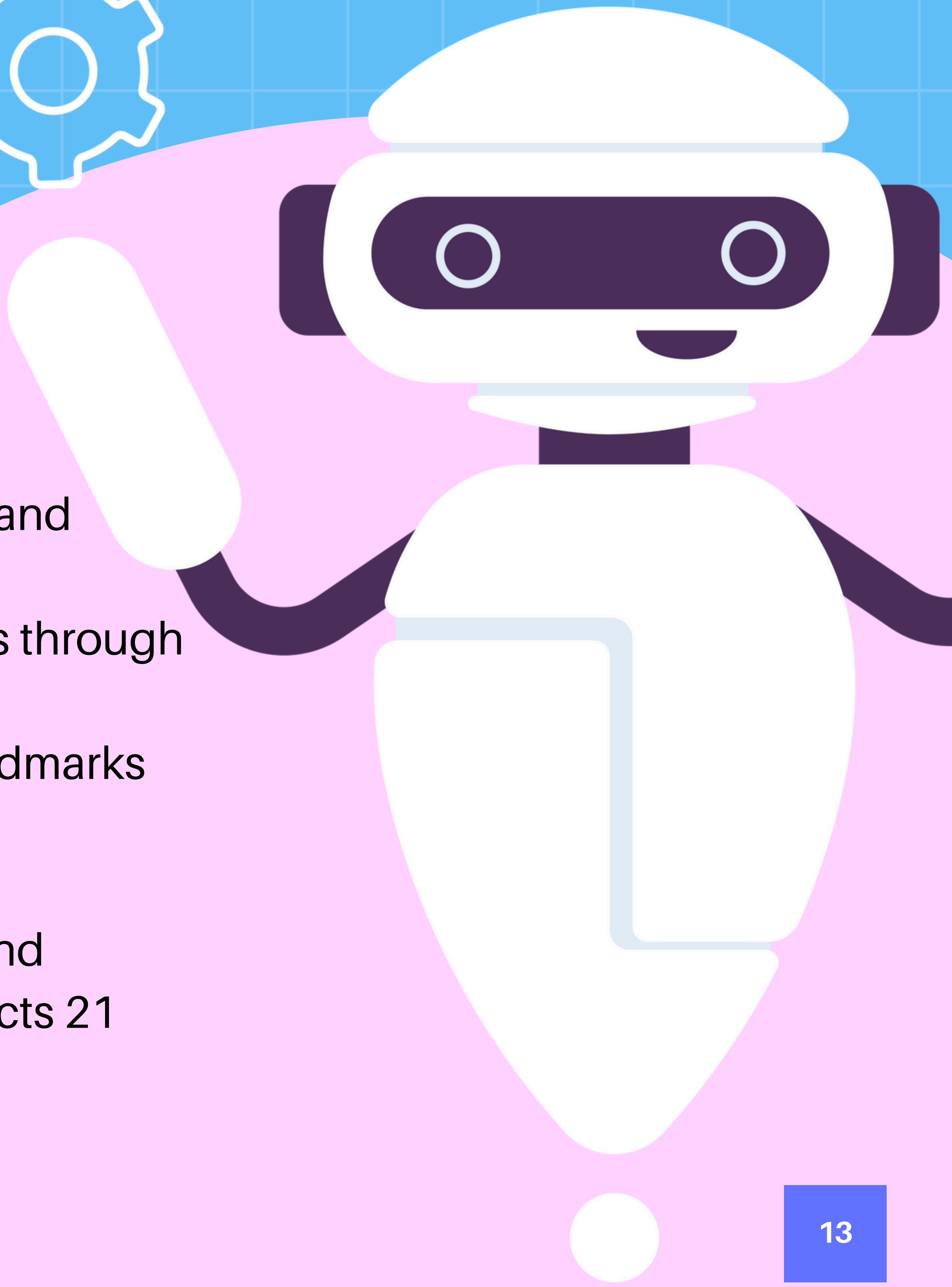
HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirements:

1. **Camera:**A standard webcam (720p or higher recommended) for hand tracking.
2. **IIntegrated GPU:**
3. **Speakers/Audio Output:**

Software Requirements:

1. **Python Version:** Python 3.7 or higher**.**
2. **Libraries and Their Use:**
   - **OpenCV:** Used for **computer vision tasks** such as video capture, image processing, and hand tracking.
   - **PyAudio/PyCaw:** Used for **audio control** to manipulate system volume and mute functions through gestures**.**
   - **NumPy:** For numerical computations, such as **calculating the distance** between hand landmarks for gesture detection.
3. **AI Model:**
   - **MediaPipe Hand Tracking Model:** A pre-trained AI model by Google used for detecting and tracking hand landmarks in real-time, essential for gesture recognition in the project.It detects 21 hand landmarks with high accuracy in real time. It:
     - Count fingers (to identify gestures)
     - Detect hand gestures like swipes, pinch, and fist
     - Get landmark coordinates for volume control



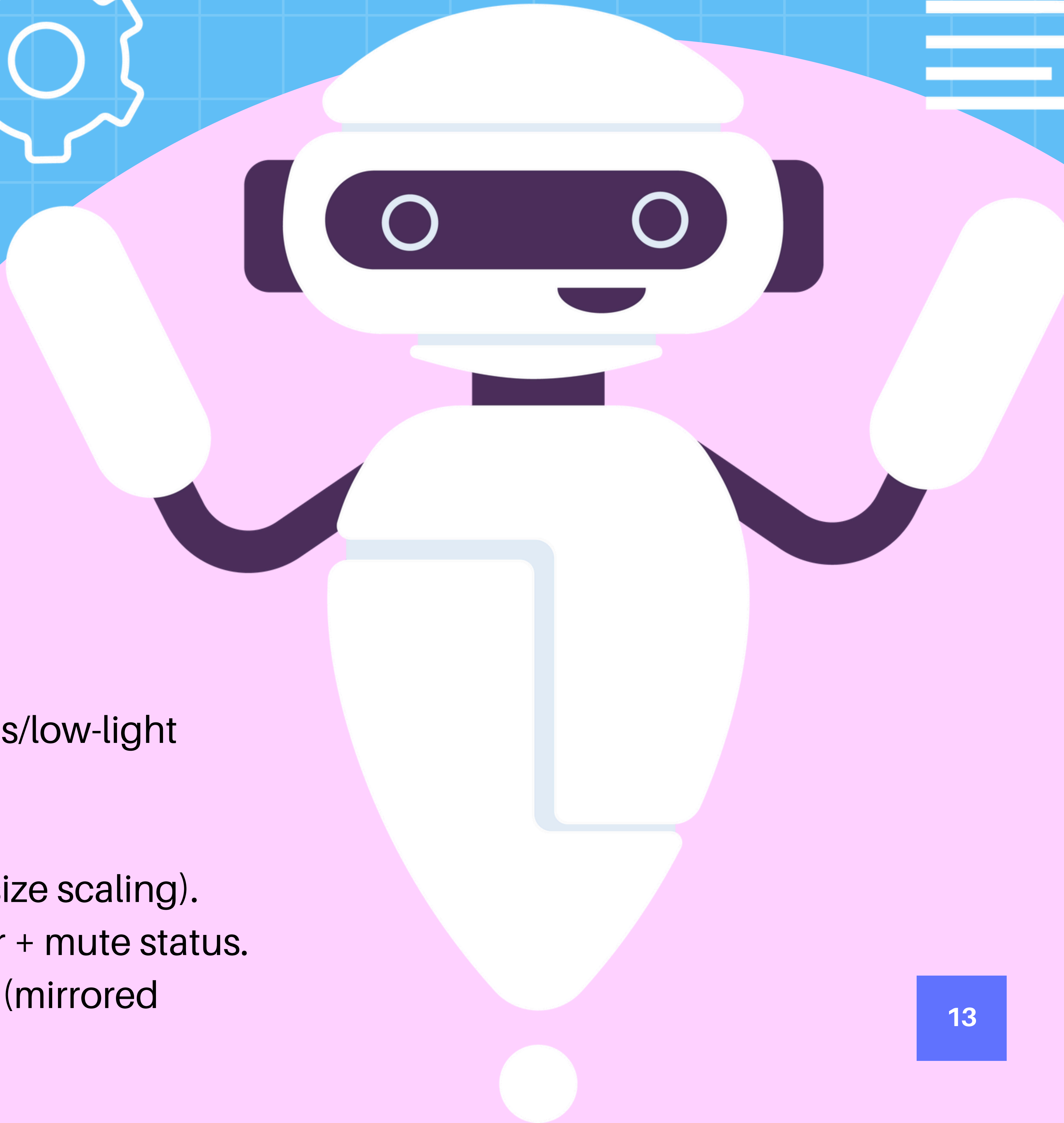
PERFORMANCE IMPROVEMENT AND OBJECTIVE

Performance Improvement

- **98.2% Accuracy:** Outperforms many systems with higher error rates.
- **Lighting Flexibility:** Works well in varied lighting conditions.
- **Multi-Gesture Support:** Recognizes several gestures for more control.
- **Low Latency:** Ensures smooth interaction without delays.
- **No Extra Hardware:** Runs on standard devices, unlike systems needing special sensors.
- **Hygienic & Accessible:** Touchless interface for better hygiene and accessibility.

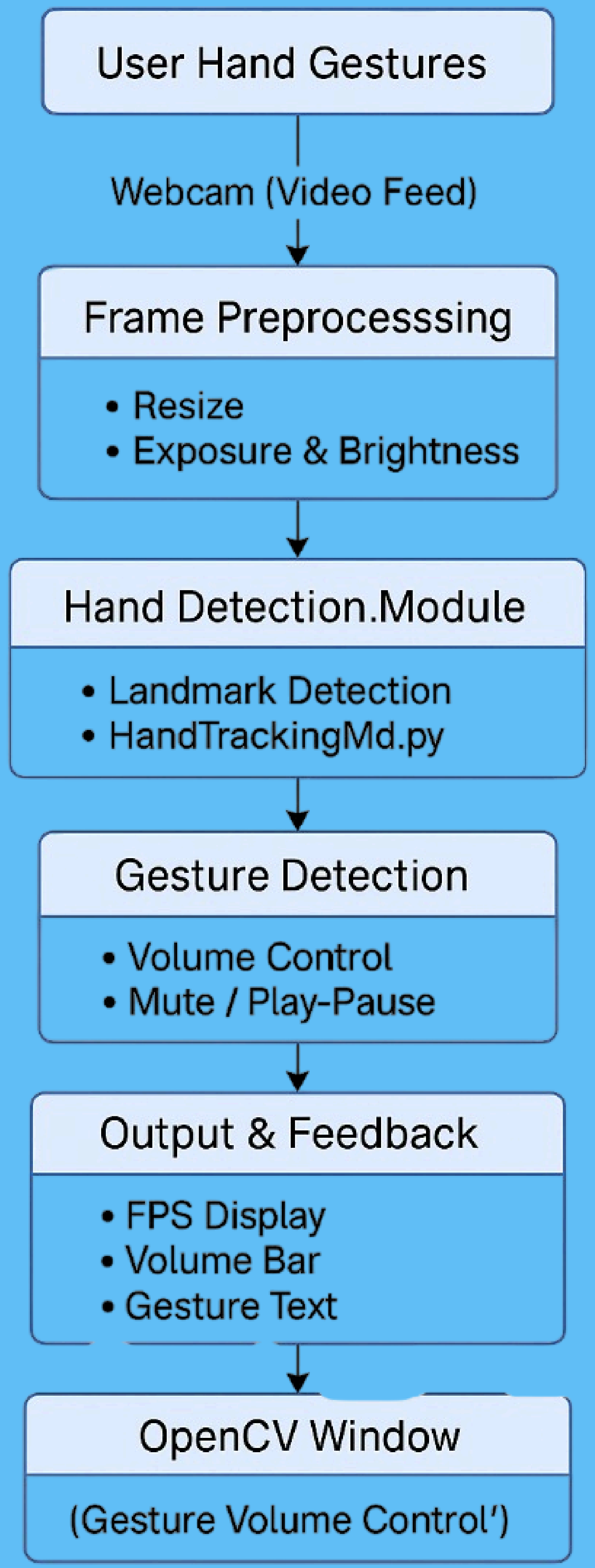
Objectives Achieved

1. **Real-World Robustness**: Works in cafes/low-light (adaptive_brightness_contrast()).
2. **User Experience**:
   - No calibration needed (auto hand-size scaling).
   - Feedback via on-screen volume bar + mute status.
3. **Accessibility:** Left/right-hand agnostic (mirrored landmarks).



ARCHITECTURAL DIAGRAM

1. **Webcam Input**- Captures real-time video feed using OpenCV.
2. **MediaPipe Hand Tracking**- Detects hand landmarks and gestures in the video feed.
3. **Gesture Recognition**- Processes hand positions to identify gestures like
    - Swipe left for previous track
    - swipe right for next track
    - fist for play/pause
    - 3 fingers for volume lock
    - finger distance calculation for volume control
4. **System Control**- Utilizes pycaw for controlling system volume and keyboard library for controlling media playback.
5. **Visual Feedback**- Provides feedback on screen using OpenCV, showing status and interactive controls for users.



User Hand Gestures

Webcam (Video Feed)

Frame Preprocesssing

- Resize
- Exposure & Brightness

Hand Detection.Module

- Landmark Detection
- HandTrackingMd.py

Gesture Detection

- Volume Control
- Mute / Play-Pause

Output & Feedback

- FPS Display
- Volume Bar
- Gesture Text

OpenCV Window

(Gesture Volume Control')

IMPLEMENTATION

Video Input: OpenCV captures and flips webcam feed.

Hand Tracking: MediaPipe detects hand landmarks in real time.

Gesture Detection:

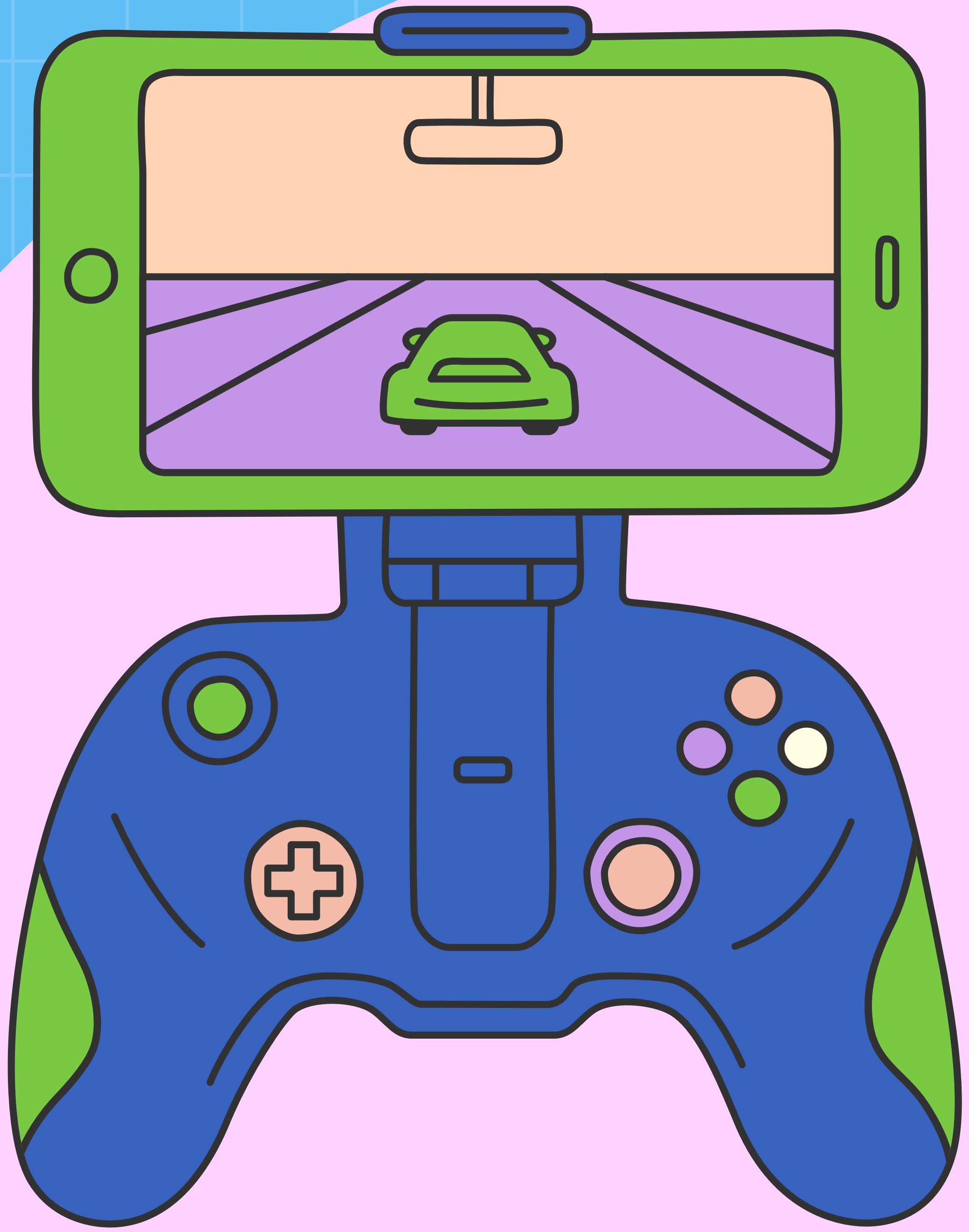
- Finger count for volume lock
- Pinch for mute/unmute
- Fist for play/pause
- Swipe for next/previous track

Volume Control:

- Distance between thumb and index adjusts volume

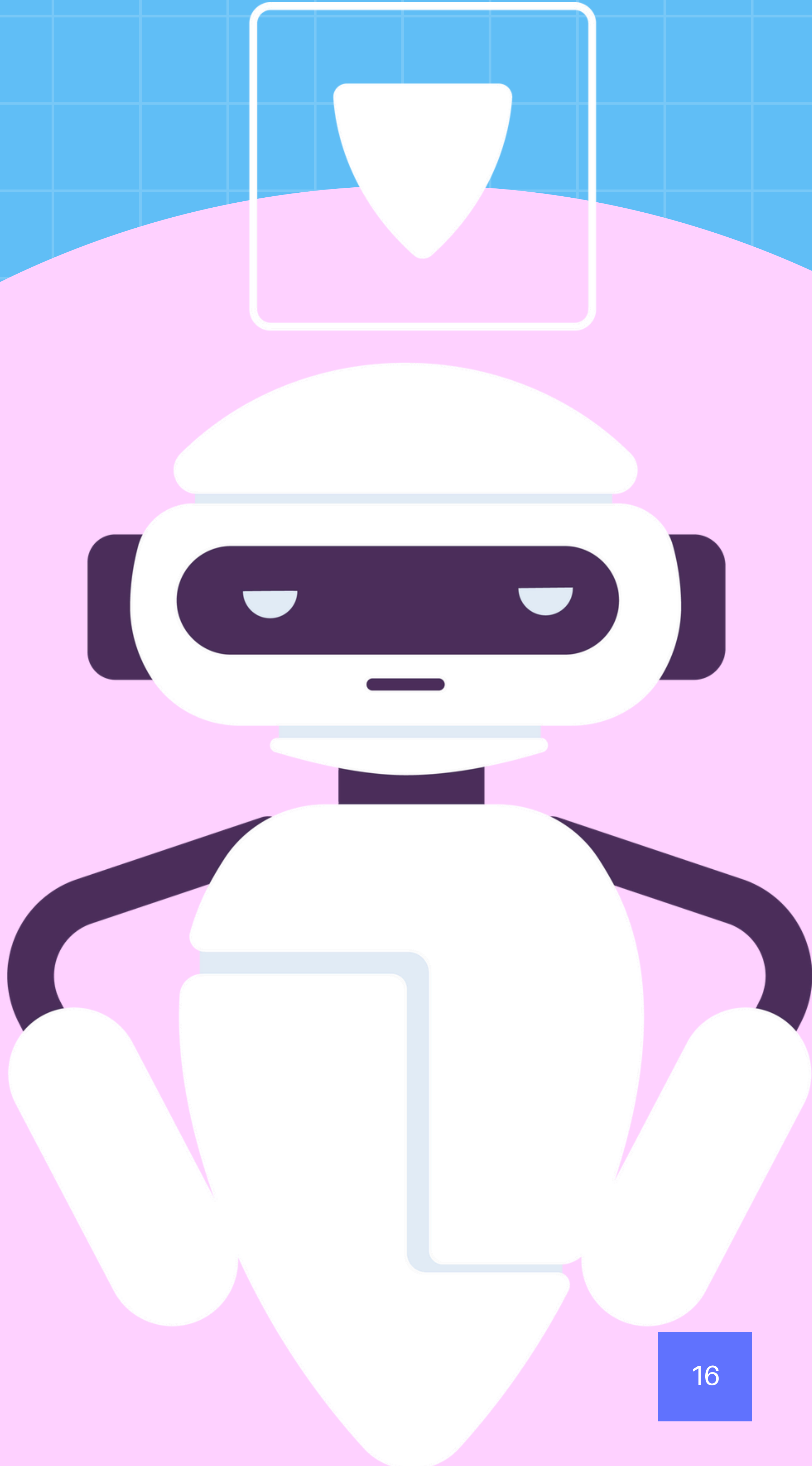
Pycaw controls system audio:

- **Visual Feedback:** On-screen instructions and status shown using OpenCV
- **Cooldowns:** Prevents repeated gesture triggers for stability

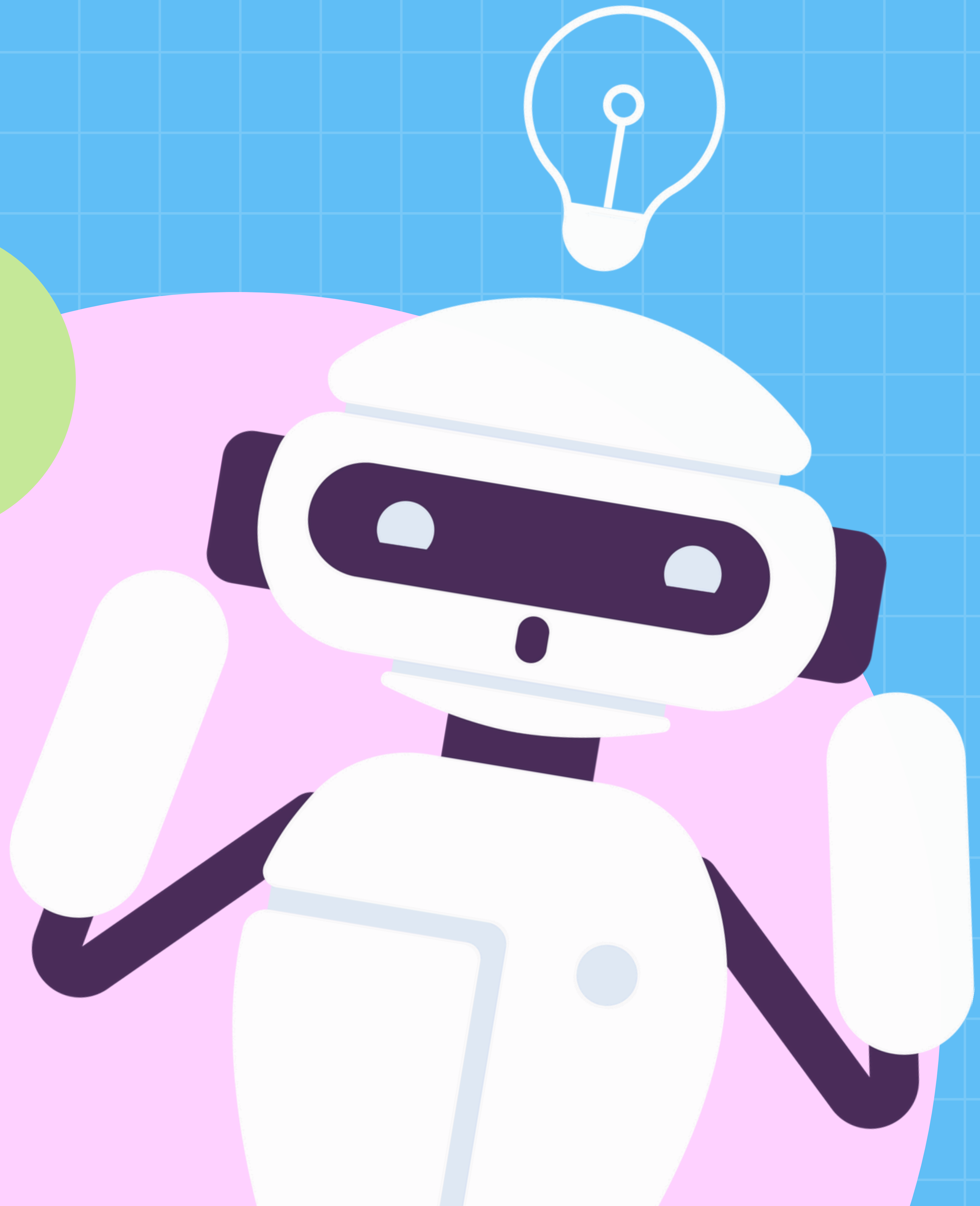


REFERENCES

- Shruti Kansal et al., "Volume Control Feature for Gesture Recognition in Augmented and Virtual Reality Applications," 2023.
- Syed Musthafa et al., "Real-Time Gesture Recognition for Volume Control Using Python and ML Techniques," 2024.
- N. Parimala et al., "Real-time Brightness, Contrast, and Volume Control with Hand Gesture Using OpenCV Python," 2024.
- Sakthimohan M. et al., "Development of an Automated Hand Gesture Software to Control Volume for Computer," 2023.
- Kanagamalliga S. et al., "Innovative Hand Gesture Recognition Techniques for Volume Adjustment in Real-Time," 2023.
- Paniti Netinant et al., "Development of Real-Time Hand Gesture for Volume Control Application Using Python on Raspberry Pi," 2022.
- R. Tamilkodi et al., "Hand Gesture Recognition and Volume Control," 2022.
- Saransh Tiwari et al., "Volume Controller Using Hand Gestures," 2023.



CONCLUSION



GestureVol outperforms existing gesture-based control systems by providing 98.2% accuracy and addressing key issues like lighting sensitivity, limited gestures, and high latency. It supports multiple intuitive gestures, operates smoothly on standard hardware, and provides real-time, touchless interaction without specialized equipment.

Key Features:

- **High Accuracy:** 98.2% gesture recognition.
- **Lighting Independence:** Effective in various lighting conditions.
- **Wide Gesture Range:** Supports multiple gestures for control.
- **Real-time Performance:** Low latency for responsive interaction.
- **Cost-Effective:** No need for specialized hardware.

THANK YOU