Computer Vision for Digital Signage Interaction

# 📌 What Is It About?

🔍 Topic:  
Using computer vision to detect and interpret hand gestures or body movements so that users can interact with digital signage (e.g., advertising screens or kiosks).  
Imagine you're walking past a smart screen in a mall, and you wave your hand — the screen responds by showing you product details or switching to a different page. No touch required!

🧠 Goal:  
Make advertisements and informational displays more engaging and interactive by:  
- Tracking gestures (like wave, thumbs up/down, point, swipe)  
- Interpreting them as commands  
- Updating the screen content accordingly

# 🧪 What You Can Build (Ideas):

Hand Gesture Detection System:  
- Detect gestures like: wave, thumbs up, swipe left/right.  
- Use these gestures to navigate menus, select options, or trigger actions on the screen.  
  
Interactive Kiosk Prototype:  
- A simple display with buttons that users "press" using gestures.  
Example: Swipe hand to scroll through product images.

# ⚙️ How to Do It (Tech Stack)

1. Data:  
- Public datasets may not exist specifically for this.  
- Use or create custom gesture data:  
 - Record short videos or frames of common gestures.  
 - Use MediaPipe, OpenCV, or YOLO to collect landmarks or track hands.  
- Example datasets:  
 - Jester Dataset (hand gestures)  
 - Kinect Leap Gesture

2. Modeling & Detection:  
- Use MediaPipe Hands or OpenCV + CNN to detect hand landmarks.  
- Use models to classify gestures (can be simple with few gestures):  
 - CNN or LSTM for video sequences.  
 - KNN/SVM for classification using landmark vectors.

3. Interaction Logic:  
- Map gestures to UI actions:  
 - Wave → Next Ad  
 - Thumbs up → Like  
 - Swipe left → Previous  
 - Point → Select

4. Interface (Optional):  
- Create a small UI or simulated screen using:  
 - Tkinter (Python GUI)  
 - Pygame  
 - Web interface (HTML/JS) connected to your backend

# ✅ Project Workflow

Step What to Do  
1. Define Gestures Pick 3–5 gestures (wave, swipe, thumbs up, point)  
2. Collect Data Record or synthesize video/image data  
3. Preprocess & Label Extract hand keypoints (MediaPipe), normalize, and label gestures  
4. Train Classifier Train a model to recognize gestures  
5. Build Interface Link recognized gestures to actions on a display or kiosk mockup  
6. Test Interaction Simulate interactions to test performance and responsiveness

# 🚀 Tools & Libraries

Task Tools/Libraries  
Hand Tracking MediaPipe, OpenCV  
Gesture Recognition TensorFlow, PyTorch  
Data Recording Webcam + OpenCV  
UI Interaction Tkinter, Pygame, Web Tech

# 💡 Bonus Ideas

- Add voice commands alongside gestures.  
- Use face detection for user attention tracking.  
- Support multi-user interactions.