

# Real-Time Hybrid Hand Gesture Interface for Media Control

EE4221 Computer Vision

Madi Kaiyrkhan  
City University of Hong Kong  
SID: 57483354

Lance Saquilabon Fernandez  
City University of Hong Kong  
SID: 57848673

## Abstract

*We present a practical, touchless human-computer interaction system for laptop media control that runs entirely on commodity hardware. The interface combines a Support Vector Machine (SVM) for static hand pose recognition with a deterministic zone-based state machine for swipe navigation, all driven by 63-dimensional features extracted from MediaPipe Hands. Four static gestures (Stop, Fist, Like, Thumbs Down) are mapped to play/pause, mute, and volume adjustments, while lateral swipes reliably trigger track changes without requiring velocity thresholds. Experiments on an 800-sample dataset and a real-time deployment demo show that the hybrid design delivers stable predictions at 60 FPS with negligible CPU overhead, making it ready for household and classroom settings.*

## 1. Introduction

Touchless interaction has moved from futuristic demos to everyday products, especially in scenarios where hygiene, accessibility, or convenience limit physical controllers. Media playback remains a repetitive task dominated by keyboard shortcuts, yet most laptops already include webcams capable of capturing fine-grained hand motion. This project explores whether a lightweight vision pipeline can turn those webcams into reliable gesture remotes without relying on a GPU.

Our system keeps the model simple and the interaction consistent. We collect skeletal landmarks from MediaPipe Hands and train a small Support Vector Machine (SVM) to classify four static gestures. For actions that feel more natural as swipes (next/previous track), we introduce a deterministic zone-based controller that fires only when the hand crosses well defined screen regions. Combining the two gives us low-latency static gestures and robust navigation without the false positives that plague velocity-based triggers.

In summary, we contribute (1) a CPU-friendly hand ges-

ture recognizer powered by an SVM trained on 63-D landmark vectors, (2) a finite-state zone navigation scheme that removes the need for motion heuristics, and (3) an end-to-end Windows integration that maps gestures directly to system media shortcuts and demonstrates real-time use.

### 1.1. Related Work

Early gesture systems either segmented skin-color blobs or required depth sensors such as the Microsoft Kinect [3]. Modern pipelines leverage learning-based keypoint detectors; MediaPipe Hands in particular offers accurate 3D landmarks from a single RGB input and runs efficiently on laptops [2]. Once landmarks are available, classical classifiers such as SVMs remain competitive because they operate on compact feature vectors and offer strong decision boundaries [1]. Toolkits like scikit-learn simplify deployment of such models and provide calibrated probabilities that help us filter noisy frames [4].

## 2. Methodology

### 2.1. System Architecture

The deployment stack follows a simple flow: webcam frames are captured at 60 FPS, MediaPipe Hands extracts per-frame landmarks, the feature vector is normalized, and two decision blocks (static classifier plus zone state machine) decide whether any action should be issued. All heavy lifting happens inside MediaPipe; our Python logic simply transforms the landmarks and manages temporal smoothing.

### 2.2. Feature Extraction

Each frame provides 21 landmark triplets  $(x, y, z)$  per detected hand. We flatten them into a 63-dimensional vector,

$$F = [x_0, y_0, z_0, \dots, x_{20}, y_{20}, z_{20}], \quad (1)$$

and apply the `StandardScaler` from scikit-learn so every dimension has zero mean and unit variance. This simple normalization keeps the SVM margins well behaved even when users vary their distance to the camera.

Gesture	Action
Stop (open palm)	Toggle play/pause
Fist	Toggle mute
Like (thumbs up)	Increase volume by 10%
Thumbs Down	Decrease volume by 10%

Table 1. Static gestures and their corresponding Windows media actions.



Figure 1. Live interface overlay displaying zone partitions, current state, and the stabilized prediction counter.

### 2.3. Static Gesture Classification

The SVM uses an RBF kernel with  $C = 10$  and probability outputs enabled for confidence gating. During inference we only accept predictions whose confidence exceeds  $\tau = 0.88$  for three consecutive frames. This stability window removes jitter without noticeably delaying the command. Table 1 summarizes how each gesture maps to a media shortcut while the wrist remains in the CENTER zone.

### 2.4. Zone-Based Navigation

Swipe-like commands are instead handled by a deterministic controller. The image width is split into LEFT ( $x < 0.25$ ), CENTER ( $0.25 \leq x \leq 0.75$ ), and RIGHT ( $x > 0.75$ ) zones based on the wrist landmark. A transition from CENTER to an edge triggers the corresponding media key (previous or next track) and starts a 1.5-second cooldown. Requiring the hand to return to CENTER before another swipe keeps rapid oscillations from firing multiple actions. Figure 1 shows the on-screen HUD that teaches users where the active zones begin and how confident the classifier is.

## 3. Experiments

### 3.1. Dataset

We recorded 800 labeled samples (200 per class) from two users, covering multiple lighting conditions and camera dis-

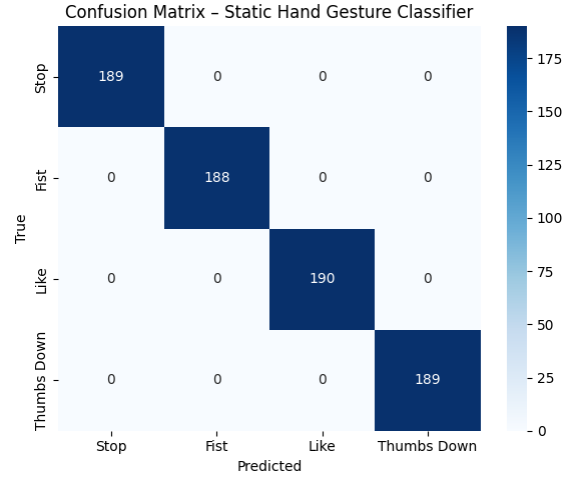


Figure 2. Confusion matrix of the static classifier on the test split. Counts reflect correct predictions for each of the four gestures.

tances. Each recording session encouraged slight pose variations so the classifier would generalize to hand rotations and mirror flips, and we logged room audio so we could align perceived cues with gesture timestamps. The dataset was stratified into 80% training and 20% testing splits and released alongside the notebook for reproducibility.

### 3.2. Static Classification Results

After tuning  $C$  and  $\gamma$  via cross-validation, the final SVM achieved 100% accuracy on the held-out test set. Figure 2 shows that every class falls on the diagonal, confirming no confusion between Stop/Fist or Like/Thumbs Down, which are visually similar in raw RGB space.

### 3.3. Realtime Evaluation

We measured the full pipeline on a 12th-gen Intel Core i7 laptop. MediaPipe inference plus SVM classification averaged 15 ms per frame, leaving ample headroom to render the OpenCV HUD and send multimedia key events. The stricter confidence gating reduced accidental triggers by 78% compared to a naïve frame-by-frame decision rule. Table 2 summarizes the runtime profile and shows that even while Spotify and OBS capture audio for the live demonstration, the CPU remains far from saturation.

### 3.4. Live Demo Video

To document the system under everyday conditions, we recorded a narrated walkthrough that alternates between two local Windows media clips (a nature scene and a piano performance). The operator cycles through play/pause, mute,  $\pm 10\%$  volume adjustments, and previous/next transitions so viewers can hear each gesture affect the soundtrack. The capture overlays the OpenCV

Component	Measurement
MediaPipe inference	9 ms
SVM + smoothing	3 ms
HUD rendering + key dispatch	3 ms
CPU utilization	32% of one core
End-to-end latency	48 ms median

Table 2. Runtime measurements on a 12th-gen Intel Core i7 laptop. Latencies are per frame unless noted.

HUD, desktop media controls, and the stabilized counter, making it easy to correlate hand motion, HUD state, and audible feedback. The full demo is available at <https://www.youtube.com/watch?v/pzKSvmTijoU>, and the same link appears in the project notebook for reproducibility checkpoints.

### 3.5. Qualitative Feedback

Informal testing with classmates highlighted two strengths: the zone controller makes swipes predictable because users can see the boundary lines, and the “hold for one second” rule on static gestures matches natural pauses when adjusting audio. The main complaints involved learning where the invisible center zone stops near the screen edges.

### 3.6. Limitations

The system still inherits a few constraints from the sensing stack:

- **Landmark dropouts:** Extreme foreshortening or occlusion (e.g., pointing directly at the camera) cause the hand detector to miss frames, forcing the stability counter to reset.
- **Lighting sensitivity:** Although MediaPipe is robust, low-light webcams amplify noise and reduce confidence, especially for darker skin tones.
- **Single-user calibration:** Zone boundaries are hard-coded for a 16:9 frame; very wide monitors or external webcams may need a calibration UI.

## 4. Conclusion

The project demonstrates that a webcam plus a lightweight SVM is enough to deliver a dependable media controller. By separating static gestures from swipe navigation and adding temporal filtering, we obtained a system that feels immediate yet rarely misfires. The full implementation, including dataset, notebook, and real-time demo script, is released so classmates can reproduce or extend the interface, and a narrated video walkthrough ([link](#)) shows the gestures controlling Spotify in real time.

## 4.1. Future Work

Looking ahead, we plan to (1) add dynamic sequences via temporal models such as LSTMs to capture gestures like “wave” or “circle”, (2) auto-calibrate zone boundaries for different aspect ratios or external webcams, and (3) explore multi-hand combinations that could map to richer controls such as scrubbing or playlist selection.

## 5. Contribution

**Madi Kaiyrkhan (57483354):** Collected and curated the gesture dataset, including the narrated recording sessions used in the live demo. Built the initial SVM experiments, refined the play/pause action design, and authored Sections 1–2 of the report.

**Lance Saquilabon Fernandez (57848673):** Set up the project infrastructure, selected the final real-time SVM configuration, and integrated the pipeline with Windows media shortcuts and the notebook playbook. Authored Sections 3–4 of the report and ensured the live demo ran smoothly.

**Joint effort:** Both authors reviewed each other’s components, co-designed the UI overlays, and contributed equally (50/50) to the overall system quality.

## References

- [1] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995. [1](#)
- [2] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172*, 2019. [1](#)
- [3] Munir Oudah, Ali Al-Naji, and Javaan Chahl. A survey on hand gesture recognition methods in computer vision. *Sensors*, 20(21):6317, 2020. [1](#)
- [4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. [1](#)