International Journal of Science, Strategic Management and Technology
Volume **02** Issue **05** May-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Gesture Vision: Gesture Controlled Virtual Mouse

[2]Arjun V,
[3]Kubendran. D,
[4]Logeshwaran. M
UG Scholar,
Vels Institute of Science,
Technology and Advanced Studies (VISTAS),
Pallavaram, Chennai - 600117,
Tamil Nadu, India.
[2] ar.juna3175@gmail.com
[3] kubendran298@gmail.com
[4] logeswaran2005loges@gmail.com

[1]A. S. Arunachalam
1Professor,
Vels Institute of Science,
Technology and Advanced Studies (VISTAS),
Pallavaram, Chennai - 600117,
Tamil Nadu, India.
[1] arunachalam1976@gmail.com

https://doi.org/10.55041/ijsmt.v2i5.012
Cite this Article: V, A., D, K. & M, L. (2026). Gesture Vision: Gesture Controlled Virtual Mouse. International Journal of Science, Strategic Management and Technology, *02*(05). https://doi.org/10.55041/ijsmt.v2i5.012

Abstract—The proliferation of touchless human–computer interaction methods has motivated the development of vision-based cursor control systems requiring no physical hardware. This paper presents Gesture Vision, a complete Gesture Controlled Virtual Mouse (GVVM) system that enables real-time cursor manipulation and mouse action execution through hand gestures captured via a standard webcam. The system employs Google's MediaPipe Hands framework for 21-landmark hand tracking, a rule-based gesture classification engine mapping ten distinct finger configurations to mouse actions, and a One-Euro filter-based cursor smoothing pipeline that eliminates jitter and enables fluid pointer movement by adapting its cutoff frequency to signal velocity. Implemented in Python using OpenCV and PyAutoGUI, the system achieves sub-30ms end-to-end inference latency at 30 FPS on commodity CPU hardware. The system integrates an advanced eye-tracking module using MediaPipe FaceMesh with single left-iris tracking, an EMA-based gaze range learning algorithm, and adaptive EAR baseline calibration for robust blink-triggered clicking. A voice command controller supports over 50 spoken commands — including cursor movement, window management, clipboard, and system operations — with both online (Google Speech API) and fully offline (PocketSphinx) recognition modes. The complete application is deployed as a lightweight desktop executable compatible with Windows, Linux, and macOS, requiring only a standard webcam and no specialised hardware or trained neural network at inference time.***Keywords***—Gesture recognition, virtual mouse, MediaPipe Hands, hand landmark detection, human– computer interaction, OpenCV, PyAutoGUI, One-Euro filter, eye tracking, voice control, touchless control, computer vision, real-time systems, FaceMesh, blink detection.

I. INTRODUCTION

The emergence of touchless human–computer interaction has transformed the landscape of accessibility technology and natural user interfaces. Traditional peripheral devices such as the mouse impose physical constraints and limit usability for individuals with motor impairments, in sterile or clinical environments, and in kiosk or embedded system contexts. Vision-based gesture recognition systems offer a compelling alternative by interpreting natural hand movements as control signals without requiring any specialised sensor hardware beyond a standard webcam.

Recent advances in hand tracking frameworks, particularly Google's MediaPipe Hands — which provides real-time 21-landmark hand skeleton detection on commodity CPU hardware — have substantially lowered the barrier to deploying gesture-based interfaces. However, the majority of existing gesture-controlled mouse systems are either

D. Cursor Smoothing and Eye Tracking

One-Euro Filter for Cursor Smoothing. The cursor controller applies a One-Euro filter [9] independently on the X and Y cursor coordinates. The One-Euro filter adapts its low-pass cutoff frequency as a linear function of the estimated signal derivative (velocity): at low velocity the cutoff is low, applying strong smoothing to eliminate jitter; at high velocity the cutoff rises, reducing smoothing lag during fast gestures. The filter is parameterised with min_cutoff=1.5 and beta=0.03 for hand tracking, and min_cutoff=0.8 and beta=0.007 for iris-based eye tracking. The tighter eye-tracking parameters reflect the lower signal bandwidth of deliberate gaze shifts compared to fast hand movements.

Eye Tracking Module. The eye tracking module uses MediaPipe FaceMesh (468 landmarks, refine_landmarks=True) to compute iris position ratios

limited to a single action (cursor movement), require depth sensors, rely on computationally expensive deep learning classifiers, or are restricted to controlled laboratory lighting conditions.

This paper presents Gesture Vision (v3): a complete, deployable Gesture Controlled Virtual Mouse system that addresses these limitations. The system supports ten distinct gesture-to-action mappings through a rule-based classification engine, integrates an advanced eye-tracking mode using MediaPipe FaceMesh for single left-iris-based cursor control and blink-triggered clicking, and incorporates a comprehensive voice command module supporting over 50 spoken commands. The One-Euro filter
[9] is applied for velocity-adaptive cursor smoothing, and the system achieves sub-30ms end-to-end latency at 30 FPS on standard consumer hardware with no trained neural network at inference time.

II. LITERATURE REVIEW

Mitra and Acharya (2007) conducted a foundational survey of gesture recognition methods, categorising approaches into static posture recognition and dynamic motion recognition, highlighting the trade-off between model complexity and real-time performance [2]. Their work established that rule-based geometric methods offer superior computational efficiency for constrained hardware compared to data-driven approaches.

Rautaray and Agrawal (2015) reviewed vision-based hand gesture recognition systems for HCI across a decade of research, identifying lighting sensitivity, background clutter, and finger occlusion as the primary failure modes of monocular RGB systems [8]. Their analysis confirmed that landmark-based approaches consistently outperform contour and colour segmentation methods in unconstrained environments.

Lugaresi et al. (2019) introduced MediaPipe, a cross-platform framework for building real-time perception pipelines, and subsequently Zhang et al. (2020) demonstrated MediaPipe Hands achieving 21-landmark hand detection at 30+ FPS on mobile CPU hardware using a two-stage palm-detector and landmark-regression pipeline [1, 5]. This framework forms the core hand detection component of the proposed GVVM system.

Hasan and Mishra (2012) demonstrated that inter- landmark Euclidean distance and finger extension angle are sufficient discriminative features for a vocabulary of five or more distinct gestures with accuracy exceeding 90% under controlled conditions [6]. Their geometric feature analysis directly informed the finger extension rules adopted in this system.

Casiez, Roussel, and Vogel (2012) introduced the One- Euro filter — a simple, parameter-tunable low-pass filter for noisy pointer signals that adapts its cutoff frequency to the derivative (speed) of the input signal [9]. At low speeds the filter applies strong smoothing to eliminate jitter; at

within the left eye bounding box. Single-iris tracking (left iris only) is used for cursor control. A gaze range learning algorithm initialises over 90 frames using exponential moving average (EMA, alpha=0.08) to gradually expand the tracked gaze bounds. A separate iris-open EAR gate (80% of the adapted EAR baseline) validates iris ratio readings independently of the blink threshold. Blink detection uses the Eye Aspect Ratio (EAR) with a 30-frame adaptive baseline. A quick left blink triggers left click; a sustained left blink (≥0.35s) triggers double click; a quick right blink triggers right click.

E. Voice Command Controller and Screen Action Execution

Voice Command Module. A voice command module running in a dedicated daemon thread uses the SpeechRecognition library in a hybrid recognition mode: PocketSphinx offline recognition is attempted first; Google Speech API is used as a fallback. The module supports over
50 spoken commands across seven categories: cursor movement, mouse clicks, scrolling, clipboard operations, file and application control, window and tab management, and system operations.

TABLE II. VOICE COMMAND CATEGORIES (v3)

Category	Example Commands	Action Performed
Cursor Movement	"move left", "move right fast"	Move ±100 px or ±300 px (fast)
Mouse Clicks	"click", "right click", "double"	Left / right / double click
Scrolling	"scroll up", "down"	Scroll wheel ±5 units
Clipboard	"copy", "paste", "cut", "undo"	Ctrl+C/V/X/Z/Y, Ctrl+A
File / Application	"save", "screenshot", "open youtube"	Ctrl+S, screenshot PNG, browser
Window / Tab	"new tab", "minimize", "close window"	Ctrl+T, Win+M, Alt+F4
System	"task manager", "lock screen"	Cross-platform launcher

Screen Action Execution. PyAutoGUI translates classified gesture states into OS-level mouse events. Cursor movement maps the index fingertip (landmark 8) coordinates from webcam frame space to screen coordinates using linear interpolation with a 30-pixel border exclusion zone. Click and scroll actions apply a temporal debounce (350ms for clicks, 120ms for scrolls). SPREAD_DRAG bypasses debouncing to preserve smooth curved strokes. Drag operations use PyAutoGUI's mouseDown/mouseUp API. All calls are wrapped in FailSafeException handlers.

V. RESULTS AND DISCUSSION

The system was evaluated on a standard consumer laptop (Intel Core i5-10th Gen, 8 GB RAM, integrated webcam)

high speeds it reduces smoothing to prevent perceptible lag. This velocity-adaptive behaviour is applied in the proposed system for both hand-tracked and iris-tracked cursor control.

Welch and Bishop (1995) provided the theoretical foundation for Kalman filter-based state estimation [7]. While Kalman filters are optimal under Gaussian noise assumptions, their fixed noise model introduces asymptotic lag during fast gesture transitions. The One-Euro filter was selected for this system as it achieves lower perceptible lag for pointer control tasks on real webcam input.

III. PROBLEM DEFINITION

Gesture-based virtual mouse systems must satisfy four interrelated requirements simultaneously. First, the gesture classification engine must achieve reliable action discrimination across ten distinct gesture states with low false-positive rates under varying hand orientations and lighting conditions. Second, the system must maintain sub- 30ms end-to-end latency from frame capture to screen action execution, as latency exceeding 50ms is perceptible as lag and significantly degrades usability [8].

Third, cursor movement must be smooth and stable despite inherent noise in landmark coordinate estimates produced by the MediaPipe regression model. Raw landmark positions exhibit high-frequency jitter at 5–15 pixel amplitude that renders direct coordinate-to-screen mapping unusable without filtering. Fourth, the system must operate on commodity consumer hardware without GPU acceleration, specialised lighting, or external depth sensors.

A further challenge is preventing unintended action triggers. The geometric regions separating gesture states — particularly between 'cursor move' and 'left click' — must incorporate temporal debouncing to prevent spurious click events during normal cursor movement. Eye-tracking additionally requires robust separation between the blink threshold and the iris-open gate, to avoid calibration instability when the eye is partially closed.

IV. PROPOSED SYSTEM

The Gesture Vision system comprises five integrated components forming a complete pipeline from webcam frame capture to desktop mouse action execution.

A. Webcam Capture and Frame Preprocessing

OpenCV VideoCapture acquires frames at 640×480 resolution targeting 60 FPS (throttled to 30 FPS by processing time). The camera buffer is set to a depth of 1 frame (CAP_PROP_BUFFERSIZE = 1) to eliminate queued-frame lag. Each frame is horizontally flipped to correct the mirror effect, converted from BGR to RGB colour space for MediaPipe compatibility, and passed to the landmark detection pipeline.

running Windows 10. Evaluation metrics included gesture recognition accuracy, end-to-end latency, cursor smoothness, and usability across five test participants performing standardised tasks.

TABLE III. PERFORMANCE COMPARISON WITH RELATED SYSTEMS

System	Accuracy	FPS	Latency	Platform
Mitra & Acharya [2]	87.3%	18	55ms	CPU
Rautaray et al. [8]	91.2%	22	45ms	CPU
Proposed GVVM (v3)	95.8%	30+	<30ms	CPU

Gesture classification accuracy was measured across 1000 intentional gesture events (100 per gesture class across ten classes) performed by five participants under indoor fluorescent lighting. Left and right click achieved the highest precision. Scroll up and double click achieved high accuracy; scroll down and drag were reliably distinguished by the 40-pixel threshold. Copy and paste had the lowest error rates when the index finger was clearly lowered. Freeze was unambiguous with all five fingers extended.

End-to-end latency — measured from frame capture to OS mouse event — averaged 22 ms (std 4 ms) over 1000 measurement trials, well within the 50 ms perceptual threshold. The One-Euro filter reduced perceived cursor jitter compared to both the unfiltered direct-mapping baseline and fixed exponential smoothing, confirmed by participants reporting improved control precision. The velocity-adaptive cutoff mechanism provided responsive tracking during fast gestures while maintaining stability during precision pointing.

The eye tracking module was evaluated separately. After the 90-frame EMA range warmup, iris-based cursor control consistently reached all four screen corners. Blink-click accuracy exceeded 92% for deliberate single blinks. Voice command recognition latency in offline mode averaged under 200 ms per command. The hybrid online/offline architecture achieved 100% command coverage with internet and greater than 85% coverage offline.

VI. CONCLUSION

This paper presented Gesture Vision (v3), a complete Gesture Controlled Virtual Mouse system achieving real- time touchless cursor control using MediaPipe Hands landmark detection, a ten-gesture rule-based classification engine, One-Euro filter cursor smoothing, MediaPipe FaceMesh-based eye tracking with single left-iris control and adaptive EAR blink clicking, and a voice command module supporting over 50 spoken commands in hybrid online/offline recognition mode. The system achieves sub-

cv2.setUseOptimized(True) is set at startup to enable SIMD acceleration.

B. Hand Landmark Detection

MediaPipe Hands detects and tracks 21 three-dimensional landmarks per hand in real time using a two-stage model: a palm detector identifying the bounding box, followed by a landmark regression model operating on the cropped hand region. Detection is constrained to a single hand (max_num_hands=1) with minimum detection confidence 0.85 and minimum tracking confidence 0.65. The tracking confidence is set deliberately lower than detection confidence to allow the tracker to re-lock onto fast-moving hands without repeated full detections.

C. Gesture Classification Engine

A rule-based classification engine maps the 21-landmark skeleton to one of ten action states. Finger extension for the index through pinky fingers is determined by comparing the y-coordinate of each fingertip landmark against the y- coordinate of the corresponding proximal knuckle. Thumb extension uses a rotation-independent distance method: the Euclidean distance from thumb tip (landmark 4) to pinky base (landmark 17) is compared against the distance from thumb base (landmark 2) to pinky base; the thumb is extended when the tip distance exceeds 1.2× the base distance.

Ten gesture-to-action mappings are defined: (1) Index finger alone extended with thumb down maps to cursor movement. (2) Closed fist triggers left click. (3) Thumb and index both extended with separation exceeding 40 px triggers drag/draw mode. (4) The same thumb-index pair with separation below 40 px triggers scroll down. (5) Index and middle fingers together map to scroll up. (6) Three fingers map to double click. (7) Pinky alone extended maps to right click. (8) Thumb and middle map to copy (Ctrl+C).
(9) Thumb and ring map to paste (Ctrl+V). (10) Open palm toggles cursor freeze.

TABLE I. GESTURE-TO-ACTION MAPPING

Gesture	Action	Finger State
Index only (thumb down)	Move Cursor	Index up, all others down
Fist (all closed)	Left Click	All 5 fingers down
Thumb + Index spread	Drag / Draw	Thumb + index up, gap >= 40px
Pinch (thumb + index close)	Scroll Down	Thumb + index up, sep < 40px
Peace sign (I+M)	Scroll Up	Index + middle up, thumb down
Three fingers (I+M+R)	Double Click	Index+middle+ring, thuml down
Pinky only	Right Click	Pinky up, others down
Thumb + Middle	Copy (Ctrl+C)	Thumb+middle up, index down
Thumb + Ring	Paste (Ctrl+V)	Thumb+ring up, index down
Open Palm (all 5)	Freeze/Unfreeze	All 5 fingers up

30ms end-to-end latency at 30 FPS on commodity CPU hardware with no trained neural network at inference time.

The rule-based geometric classification approach provides transparent, interpretable action mappings trivially configurable without retraining. The One-Euro filter provides measurably better pointer smoothness than fixed exponential smoothing while introducing zero asymptotic lag during fast gestures. The single left-iris tracking architecture eliminates dual-iris averaging instability. The complete system is deployable as a standalone Python application on Windows, Linux, and macOS requiring only a standard webcam and no specialised hardware.

Future work includes machine learning-based gesture classification for improved accuracy under challenging lighting and skin tone variation, multi-hand support enabling two-handed interactions, configurable gesture-to- action mapping via a user-editable profile file, persistent eye calibration stored across sessions, and mobile deployment via TensorFlow Lite for Android and iOS platforms.

REFERENCES.

[1] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "MediaPipe Hands: On-device real-time hand tracking," in Proc. IEEE/CVF CVPRW, 2020.

[2] S. Mitra and T. Acharya, "Gesture recognition: A survey," IEEE Trans. Systems, Man, Cybernetics C, vol. 37, no. 3, pp. 311–324, 2007.

[3] G. Bradski, "The OpenCV library," Dr. Dobb's Journal of Software Tools, 2000.

[4] J. Flusser, T. Suk, and B. Zitova, 2D and 3D Image Analysis by Moments. Wiley, 2016.

[5] C. Lugaresi et al., "MediaPipe: A framework for building perception pipelines," arXiv:1906.08172, 2019.

[6] M. M. Hasan and P. K. Mishra, "Hand gesture modelling and recognition using geometric features: A review," Canadian J. Image Processing and Computer Vision, 2012.

[7] G. Welch and G. Bishop, "An introduction to the Kalman filter," Univ. North Carolina at Chapel Hill, Tech. Rep. TR 95-041, 1995.

[8] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: A survey," Artificial Intelligence Review, vol. 43, no. 1, pp. 1–54, 2015.

[9] G. Casiez, N. Roussel, and D. Vogel, "1€ Filter: A simple speed-based low-pass filter for noisy input in interactive systems," in Proc. ACM CHI, 2012, pp. 2527–2530.