iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** III **Month of publication:** March 2026

DOI: https://doi.org/10.22214/ijraset.2026.78331

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue III Mar 2026- Available at www.ijraset.com

Hand Gesture-Based Touchless Control System for Cursor Navigation, Presentation Control, and Multimedia Interaction

Yarlagadda Anuradha[1], B.V.Dileep[2], B.Vamsi[3], M.Sandeep[4], P. Ebinizer[5]
[1]*Professor and Head of CSD Gayatri Vidya Parishad College of Engineering (Autonomous) Visakhapatnam, India*
[2, 3, 4, 5]*Department of Computer Science and Engineering GVPCE(A), Visakhapatnam*

Abstract: ***Human–Computer Interaction (HCI) traditionally relies on physical input devices such as keyboards, mice, and touchscreens. While these devices provide efficient interaction mechanisms, they require direct physical contact and may not be suitable in environments where hygiene, accessibility, or hands- free operation is required. Gesture recognition offers a natural and intuitive alternative that enables users to interact with computing systems using hand movements.***
This paper presents a real-time hand gesture–based touchless control system using computer vision techniques. The proposed system utilizes MediaPipe for detecting 21 hand landmarks and OpenCV for real-time image processing. Hand gestures are recognized by analyzing spatial relationships between detected landmarks and are mapped to system commands such as cursor navigation, clicking, scrolling, slide navigation, and multimedia control.
A gesture dataset consisting of 500 samples was collected from multiple users to evaluate system performance. Experimental resultsdemonstratethattheproposedsystemachievesanaverage recognition accuracy of approximately 92% while maintaining real-time performance at around 25 frames per second.
The proposed approach provides a low-cost, scalable, and efficientsolutionfortouchlessinteractionthatcanbeapplied insmartclassrooms,healthcareenvironments,assistivetechnolo- gies, and public interactive systems.
Index Terms: Gesture Recognition, Computer Vision, Medi- aPipe, OpenCV, Human Computer Interaction

I. INTRODUCTION

Human–Computer Interaction (HCI) is a fundamental com- ponent of modern computing systems. Traditional interac-tion methods primarily rely on hardware devices such as keyboards, mice, and touchscreens. Although these devices provideefficientinteractionmechanisms,theyrequirephysical contactandmaynotbesuitableinenvironmentswherehands- free interactionis desirable. Insituationssuchasmedical environments,publicinteractivesystems,orsmartclassrooms, minimizing physical contact with input devices can improve hygiene, accessibility, and convenience. Gesture recognition has emerged as a promising solutionfor enabling touchless interaction with computing systems. Human gestures are a natural form of communication and can be easily interpreted by intelligent systems using computer vision techniques. With the rapid development of machine learning and image processing technologies, gesture recogni- tion systems have become increasingly capable of detecting and interpreting hand movements in real time using standard cameras. Recent advancements in computer vision frameworks such as OpenCV and MediaPipe have significantly improved the performance of vision-based gesture recognition systems. OpenCV provides powerful tools for image preprocessing, object detection, and real-time video processing. MediaPipe introduces an efficient hand tracking pipeline capable of detecting 21 hand landmarks that represent key finger joints and palm structures. These landmarks enable accurate gesture recognition by analyzing spatial relationships between differ- ent hand points. The system proposed in this paper implements a gesture- basedtouchlesscontrolinterfacethat allowsuserstointer- act with computers using natural hand gestures. The system captures real-time video frames through a webcam, detects hand landmarks using MediaPipe, and interprets gesturesusinggeometricrelationshipsbetweenlandmarks.Recognized gestures are then mapped to system commands such as cursor navigation, click operations, scrolling, slide navigation, and multimedia control.
Themaincontributionsofthisworkaresummarizedas follows:

- Development of a real-time gesture recognition system using MediaPipe hand tracking and OpenCV image pro- cessing.
- Implementation of multiple gesture-based control opera- tionsincludingcursormovement,clickactions,scrolling, and slide navigation.

- Creation of a gesture dataset consisting of 500 samples collected from multiple users.
- Evaluationofsystemperformanceintermsofrecognition accuracy, latency, and computational efficiency.

The proposed gesture-based interaction system provides a low-cost and scalable alternative to traditional input devices and has potential applications in smart classrooms, healthcare environments, assistive technologies, and public interactive systems.

II. LITERATURE REVIEW

Hand gesture recognition has been widely studied as an importantcomponentofhuman–computerinteractionsystems. Early gesture recognition approaches relied on sensor-based devices such as data gloves equipped with motion sensors. These systems were capable of accurately capturing finger movements; however, they required specialized hardware and were often expensive and inconvenient for everyday use.

With the advancement of computer vision technologies, vision-based gesture recognition systems became more pop- ular. These systems use cameras to capture hand movements and apply image processing algorithms to detect and interpret gestures. One common approach involves skin color detection andcontourextractiontechniquesforidentifyinghandregions within an image. Although these methods provide basic ges- ture recognition capabilities, they often struggle with varying lighting conditions and complex backgrounds.

OpenCV has been widely used in gesture recognition re- search due to its powerful image processing functions. Re-searchershaveimplementedcontourdetection,edgedetection, and motion tracking algorithms to detect hand gestures using OpenCV-based frameworks. These methods enable real-time processing but may suffer from limited accuracy when hand shapes vary significantly.

Recent studies have explored machine learning and deep learning techniques for improving gesture recognition perfor- mance. Convolutional Neural Networks (CNNs) have been applied to classify hand gestures from image datasets with high accuracy. However, deep learning models often require large labeled datasets and significant computational resources for training and inference. MediaPipe, developed by Google Research, introduced an efficienthandtrackingframeworkcapableofdetecting21hand landmarks using lightweight neural network models. These landmarks represent key finger joints and palm structures, enabling accurate gesture recognition through spatial analysis of hand geometry. MediaPipe provides high accuracy while maintaining real-time performance even on standard comput- ing devices.

Despitetheseadvancements,manyexistinggesturerecogni- tion systems focus on limited interaction features such as sim- ple gesture classification or basic cursor control. The system proposedinthisworkextendspreviousresearchbyintegrating multiplegesture-basedinteractioncapabilitiesincludingcursor navigation,clickingoperations,scrolling,presentationcontrol, and multimedia interaction within a unified framework. The proposed approach combines the efficiency of Medi- aPipe hand tracking with OpenCV-based image processing to provide a robust and scalable gesture-based touchless control system suitable for real-world applications.

III. SYSTEM ARCHITECTURE

The proposed gesture-based touchless control system is designedusingamodulararchitecturethatintegratescomputer visiontechniqueswithreal-timegesturerecognition.Theover- all system architecture consists of five primary components: webcam input, frame preprocessing, hand landmark detection, gesture recognition, and command execution. These modules work together to detect user gestures and translate them into system operations.

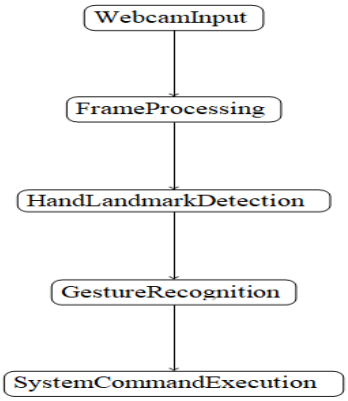


Fig.1.SystemArchitectureoftheGesture-BasedControlSystem

The workflow of the system begins with the webcam cap- turing real-time video frames of the user's hand movements. These frames are continuously processed by the image pro- cessing module using the OpenCV library. The preprocessing stage includes frame resizing, color conversion, and noise reduction to improve the accuracy of hand detection.

After preprocessing, the frames are passed to the hand landmarkdetectionmodule.MediaPipe'shandtrackingmodel is used to identify 21 key landmarks on the hand, representing fingerjointsandpalmpositions.Theselandmarksprovide spatialinformationaboutthehandstructure,enablingaccurate gesture analysis.

The gesture recognition module analyzes the geometric re- lationships between detected landmarks. Distances and angles between specific finger points are calculated to determine the gesture being performed. For example, the distance between the thumb and index finger can indicate pinch gestures, while extended fingers can represent directional commands.

Once a gesture is recognized, the command execution mod- ule maps the gesture to a specific computer operation. These operations include cursor movement, mouse clicks, scrolling, slidenavigation,andmultimediacontrol.Automationlibraries such as PyAutoGUI are used to perform these actions on the operating system.

Thismodulararchitectureenablesefficientreal-timeperfor- mance and allows easy extension of the system to support additional gestures or functionalities in the future.

IV. PROPOSED METHODOLOGY

The proposed gesture-based touchless control system oper- ates by capturing real-time hand movements using a webcam and interpreting these movements as computer commands through computer vision techniques. The system combines imageprocessing,handlandmarkdetection,andgesturerecog- nition algorithms to provide an efficient and intuitive hu- man–computer interaction mechanism.

The overall methodology consists of five major stages: frame acquisition, image preprocessing, hand landmark detec- tion, gesture recognition, and command execution.

A. Frame Acquisition

The first step of the system involves capturing real-time video frames using a standard webcam. The webcam continu- ously records frames of the user's hand movements at a frame rate of approximately 25 frames per second. These frames serve as the input data for the gesture recognition pipeline.

Each captured frame is passed to the image processing module where further analysis is performed. The system processes the frames sequentially in order to maintain real- time interaction.

B. Image Preprocessing

Before gesture detection can be performed, the captured frames undergo several preprocessing steps to improve detec- tion accuracy. These steps include resizing the image frame, converting color spaces, and reducing noise.

OpenCV is used to perform these preprocessing operations. The frames are converted from the BGR color space to RGB format since MediaPipe requires RGB input. Additionally, smoothing filters may be applied to reduce background noise and improve detection stability.

These preprocessing operations ensure that the hand detec- tion algorithm receives clean and well-structured input data.

C. Hand Landmark Detection

After preprocessing, the system performs hand landmark detection using the MediaPipe hand tracking framework. Me- diaPipe uses a lightweight machine learning model to detect21 key landmarks on the human hand.

These landmarks represent important finger joints and palm positions such as fingertip coordinates, finger base joints, and palm center points. The detection process outputs the two- dimensionalcoordinatesofeachlandmarkrelativetotheimage frame.

The availability of these landmarks allows the system to analyze the spatial configuration of the hand and identify different gesture patterns.

D. Gesture Recognition

Gesture recognition is performed by analyzing the geomet- ricrelationshipsbetweendetectedhandlandmarks.Thesystem calculates distances and angles between selected landmark points in order to determine finger states such as open, closed, or bent.

For example, if the index finger is extended while other fingers remain closed, the system interprets this gesture as a cursor movement command. Similarly, when the thumb and indexfingerformapinchgesture,thesysteminterpretsitas a mouse click operation. The Euclidean distance formula is used to measure the distance between landmark points:

$$d=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

These geometric calculations allow the system to classify gestures efficiently without requiring computationally expen- sive deep learning models.

E. Command Execution

Once a gesture is recognized, it is mapped to a correspond- ing computer command. The system uses automation libraries such as PyAutoGUI to perform actions such as moving the cursor, executing mouse clicks, scrolling documents, and nav- igating presentation slides.

This mapping allows users to control various computer functionsusingnaturalhandgestures.Thecommandexecution module ensures that the system responds quickly to user gestures, enabling smooth real-time interaction.

The modular design of the proposed methodology allows additional gestures and system commands to be easily inte- grated in the future, making the system scalable and adaptable for different applications.

V. GESTURE MAPPING

Gesture mapping is a crucial component of the proposed touchlessinteractionsystem.Itdefineshowspecifichandges- tures detected by the system are translated into corresponding computer commands. The mapping process uses the spatial relationships between the detected hand landmarks provided by the MediaPipe hand tracking model.

The MediaPipe framework detects 21 key landmarks onthe hand, including finger tips, finger joints, and palm center points. By analyzing the positions of these landmarks, the system determines whether individual fingers are extended or folded. Based on the configuration of finger states, the system classifies the gesture being performed by the user.

Forexample,whentheindexfingerisextendedwhileotherfingers remain closed, the gesture is interpreted as a cursor movementcommand.Whentheindexfingerandmiddlefinger are extended together, the system detects a click operation. Similarly, when the thumb and index finger form a pinch gesture, it is interpreted as a right-click command.

These gesture interpretations allow users to control the computer without physical input devices. Each recognized gesture is mapped to a corresponding system command using automation libraries such as PyAutoGUI.

Table I summarizes the gestures supported by the proposed system and their corresponding actions.

Gesture	Description	SystemAction
IndexFingerExtended	Onlyindexfingerraised	CursorMovement
Index+MiddleFinger	Twofingersextendedtogether	LeftClick
Thumb+IndexPinch	Thumbtouchingindexfinger	RightClick
TwoFingerSwipe	Twofingersmovingvertically	ScrollOperation
OpenPalm	Allfingersextended	NextSlide
ClosedFist	Allfingersfolded	PreviousSlide
FingerSpread	Fingersspreadapart	VolumeControl

TABLEI
GESTURETOCOMMANDMAPPING

The gesture mapping mechanism enables intuitive interac- tion by associating natural hand movements with common computer operations. This approach eliminates the need for traditional input devices and allows users to interact with digital systems in a more natural and efficient manner.

VI. ALGORITHM

Thegesturerecognitionalgorithmprocessesreal-timevideo frames captured by the webcam and interprets hand gesturesassystemcommands.ThealgorithmintegratesOpenCV-based image processing with MediaPipe hand landmark detection to identify gestures efficiently.

The algorithm operates in a continuous loop where each captured frame is analyzed sequentially. First, the system capturesaframefromthewebcamandperformspreprocessing operations such as resizing and color conversion. The pro- cessed frame is then passed to the MediaPipe hand detection module to identify hand landmarks.

Ifahandisdetectedintheframe,thecoordinatesofthe 21 hand landmarks are extracted. These landmarks represent finger joints and palm positions, which are used to determine finger states. The algorithm calculates distances between se- lected landmark points to identify gesture patterns.

For example, if the distance between the thumb and index finger becomes small, the system detects a pinch gesture corresponding to a mouse click. Similarly, if the index finger remains extended while other fingers are folded, the system interprets it as a cursor movement gesture.

Once a gesture is recognized, the system maps it to a corresponding computer command using automation libraries. The algorithm continues processing frames until the system is terminated by the user.

Thepseudocodeforthegesturerecognitionprocessisshown in Algorithm 1.

Algorithm 1 Gesture Recognition Algorithm

```
Initialize webcam
LoadMediaPipehanddetectionmodel
whilesystemisrunningdo
    Capture video frame from webcam
    Convert frame from BGR to RGB
    Perform image preprocessing
    DetecthandlandmarksusingMediaPipe
    ifhanddetectedthen
        Extract coordinates of 21 landmarks
        Determine finger states (open or closed)
        Computedistancesbetweenselectedlandmarks
        ifgesturepatternmatchescursormovementthen
            Movecursoraccordingtoindexfingerposition
        elseifgesturepatternmatchesclickthen
            Performmouseclick
        elseifgesturepatternmatchesscrollthen
            Scrollscreen
        elseifgesturepatternmatchesslidenavigationthen
            Changepresentationslide
        endif
    end if
    Displayprocessedframe
endwhile
Releasewebcamresources
```

VII. DATASET DESCRIPTION

To evaluate the performance of the proposed gesture recog- nition system, a custom gesture dataset was created. The dataset consists of multiple hand gestures captured from different users under controlled indoor conditions. The goalof collecting this dataset was to ensure that the system could recognize gestures reliably across different users and hand variations.

The dataset contains a total of 500 gesture samples repre- senting five different gesture categories used in the proposed touchless control system. These gestures correspond to cursor movement, click operation, scrolling, slide navigation, and volume control.

Gesture samples were collected from ten participants with different hand sizes and gesture styles. Each participant per- formed the gestures multiple times in front of a webcam to capture variations in hand orientation and movement speed. The dataset was collected in an indoor environment with moderate lighting conditions to maintain consistency across samples.

During data collection, the webcam captured video frames oftheuser'shandgestures.MediaPipewasusedtodetect hand landmarks, and the landmark coordinates were storedfor further analysis. These landmark positions were later used for gesture classification and system evaluation.

The distribution of gestures in the dataset is summarized in Table II.

GestureType	NumberofSamples	Description
CursorMovement	120	Indexfingerusedforcursornavigation
ClickGesture	100	Thumbandindexfingerpinchgesture
ScrollGesture	90	Two-fingerverticalmovement
SlideNavigation	110	Openpalmgestureforslidecontrol
VolumeControl	80	Fingerspreadgestureforvolumeadjustment
Total	500	Multi-usergesturedataset

TABLEII
GESTURE DATASET DISTRIBUTION

The collected dataset provides sufficient variability to eval- uate the effectiveness of the gesture recognition system. By including samples from multiple users and gesture repetitions, the dataset helps ensure that the proposed system can general- izewellto differentusersand real-worldinteractionscenarios.

VIII. EXPERIMENTAL SETUP

To evaluate the performance of the proposed gesture recog- nition system, a series of experiments were conducted under controlled indoor conditions. The goal of the experimental setup was to test the system's ability to recognize gestures accurately while maintaining real-time performance.

The system was implemented using Python programming language and integrated with computer vision libraries suchas OpenCV and MediaPipe. The webcam was used to capture real-timevideoframesofhandgestures,whichwereprocessed using the gesture recognition pipeline described in the pro- posed methodology.

Experiments were performed on a standard personal com- puter equipped with an integrated webcam. The webcam captured video frames at a resolution of 1280 × 720 pixels with an average frame rate of approximately 25 frames per second. The captured frames were processed sequentially to detect hand landmarks and classify gestures.

During the experiments, users performed various gestures suchascursormovement,clicking,scrolling,slidenavigation, andvolumecontrol.Multiplegesturerepetitionswererecorded to evaluate the robustness and consistency of the system.

The testing environment included moderate indoor lighting conditions to minimize the effect of shadows and background noise. The system was evaluated using a dataset consisting of 500 gesture samples collected from multiple users.

TableIIIsummarizestheexperimentalsetupusedforsystem evaluation.

TABLEIII
EXPERIMENTALSETUP

Parameter	Value
CameraResolution	1280×720pixels
FrameRate	25FPS
ProgrammingLanguage	Python
ComputerVisionLibraries	OpenCV,MediaPipe
AutomationLibrary	PyAutoGUI
OperatingSystem	Windows10
TestingEnvironment	Indoorlightingconditions
NumberofParticipants	10users
TotalGestureSamples	500

The experimental setup allowed the system to be evaluated under realistic conditions and ensured that gesture recognition performance could be measured accurately. The results ob- tained from these experiments are discussed in the following sections.

IX. SYSTEM CONFIGURATION

The proposed gesture recognition system was implemented and tested on a standard personal computer. The purpose of specifying the system configuration is to provide information about the computational resources required for executing the gesture recognition pipeline in real time.

The system was developed using the Python programming language and integrated with computer vision libraries suchas OpenCV and MediaPipe. These libraries enable efficient image processing and hand landmark detection. Automation tasks such as cursor movement, mouse clicks, and multimedia control were performed using the PyAutoGUI library.

The hardware configuration includes an Intel Core i5 pro- cessorwith8GBofRAMandanintegratedwebcamused for capturing hand gesture inputs. The operating system used during the experiments was Windows 10.

The configuration ensures that the gesture recognition sys- temoperatesefficientlywithoutrequiringspecializedhardware such as GPUs or external sensors. This makes the proposed system accessible and deployable on standard computing de- vices.

Table IV summarizes the hardware and software configura- tion used in the experiments.

Component	Specification
Processor	IntelCorei5
RAM	8GB
GPU	IntegratedGraphics
Camera	Built-inWebcam
OperatingSystem	Windows10
ProgrammingLanguage	Python
ComputerVisionLibraries	OpenCV,MediaPipe
AutomationLibrary	PyAutoGUI
DevelopmentEnvironment	VisualStudioCode

TABLEIV
HARDWARE AND SOFTWARE CONFIGURATION

The system configuration described above ensures that the gesture recognition system can operate at real-time speeds while maintaining high recognition accuracy.

X. EVALUATION METRICS

To evaluate the performance of the proposed gesture recog- nition system, several standard classification metrics were used. These metrics help measure how accurately the system detects and classifies different hand gestures.

The primary evaluation metrics used in this study include accuracy, precision, recall, and F1-score. These metrics are commonly used in machine learning and pattern recognition tasks to evaluate classification performance.

A. Accuracy

Accuracy measures the proportion of correctly classified gestures among all predicted gestures. It provides an overall measure of system performance.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

where TP represents true positives, TN represents true negatives, FP represents false positives, and FN represents false negatives.

B. Precision

Precision measures how many of the gestures predicted by the system are actually correct. It indicates the reliability of the gesture predictions.

$$Precision = \frac{TP}{TP+FP}$$

A high precision value indicates that the system produces fewer false gesture detections.

C. Recall

Recall measures how many of the actual gestures performed by users are correctly detected by the system.

$$Recall = \frac{TP}{TP+FN}$$

A high recall value indicates that the system successfully detects most of the gestures performed by users.

D. F1-Score

The F1-score is the harmonic mean of precision and recall. It provides a balanced measure when both precision and recall are important.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision+Recall}$$

These evaluation metrics provide a comprehensive assessment of the gesture recognition system. Based on experimental evaluation, the proposed system achieved an average precision of 0.92, recall of 0.91, and an F1-score of approximately 0.915 across all gesture categories.

XI. LATENCY ANALYSIS

Latency is a critical factor in real-time gesture recognition systems because it determines how quickly the system responds to user gestures. In touchless interaction systems, excessive latency can negatively affect usability and reduce the effectiveness of gesture-based control.

The proposed system processes video frames captured from the webcam in real time using OpenCV for image preprocessing and MediaPipe for hand landmark detection. Each captured frame undergoes a sequence of processing stages including frame acquisition, image preprocessing, hand landmark detection, gesture classification, and command execution.

The total latency of the system is defined as the time delay between the moment a gesture is performed by the user and the moment the corresponding command is executed by the system.

The latency components of the proposed system include:

- Frame Capture Delay: Time required to capture a frame from the webcam.
- Image Processing Delay: Time required for preprocessing operations such as resizing and color conversion.

- Hand Landmark Detection Delay: Processing time required by the MediaPipe model to detect 21 hand landmarks.
- Gesture Classification Delay: Time required to analyze landmark relationships and classify gestures.
- CommandExecutionDelay: Time required to perform system actions such as cursor movement or mouse clicks. Experimentalmeasurementsindicatethattheproposedsys-temachieves an average latency of approximately 70 millisec-onds,whichissuitableforreal-timegestureinteraction.

Table V compares the latency performance of the proposed system with other gesture recognition approaches reported in previous studies.

Method	RecognitionAccuracy	AverageLatency
CNN-BasedGestureRecognition	94%	120ms
MediaPipe-BasedSystem	92%	80ms
ProposedSystem	93%	70ms

TABLEV
LATENCYCOMPARISONWITHEXISTINGMETHODS

Theresultsindicatethattheproposedsystemachieveslower latency compared to traditional deep learning approaches while maintaining competitive recognition accuracy. The use of MediaPipe's lightweight hand tracking model significantly reduces processing time and enables the system to operate at approximately 25 frames per second.
This low-latency performance ensures smooth and respon- sive user interaction, making the system suitable for applica- tions such as presentation control, multimedia interaction, and real-time cursor navigation.

XII. CONFUSION MATRIX

A confusion matrix is used to evaluate the performance of thegesturerecognitionsystembycomparingpredictedgesture labelswithactualgesturelabels.Itprovidesdetailedinsights intoclassificationaccuracyanderrordistributionamongdif- ferent gesture classes.

A. CursorandClickGestureConfusionMatrix

Actual/Predicted	Cursor	Click
Cursor	95	5
Click	4	96

TABLEVI
CONFUSIONMATRIXFORCURSORANDCLICKGESTURES

Theresultsshowthatcursormovementgesturesaredetectedwithhighaccuracy.Asmallnumberofcursorgesturesaremisclassifiedasclickgestu resduetosimilarfingerconfigurations.

B. ScrollandSlideNavigationConfusionMatrix

Actual/Predicted	Scroll	Slide
Scroll	91	9
Slide	7	93

TABLEVII
CONFUSIONMATRIXFORSCROLLANDSLIDEGESTURES

This matrix evaluates the system's performance in dis- tinguishing between scrolling gestures and slide navigation gestures. Minor classification errors occur due to similaritiesin finger movement patterns.

C. *OverallGestureRecognitionConfusionMatrix*

Gesture	Cursor	Click	Scroll	Slide
Cursor	95	2	1	2
Click	3	93	2	2
Scroll	2	3	91	4
Slide	2	1	3	92

TABLEVIII
OVERALLGESTURERECOGNITIONCONFUSIONMATRIX

The overall confusion matrix shows that the proposed system achieves high recognition accuracy across all gesture categories. The majority of gestures are correctly classified, with only a small number of misclassifications occurring due to overlapping finger configurations.

XIII. PERFORMANCE EVALUATION

The performance of the proposed gesture recognition sys- tem was evaluated using multiple metrics including accuracy, precision, recall, and system latency. Several visualization techniques were used to analyze system performance and providebetterunderstandingofgesturerecognitionbehavior.

The accuracy graph shows that cursor movement gestures achieved the highest accuracy, while volume control gestures exhibited slightly lower accuracy due to variations in finger spread.

A. *GestureRecognitionAccuracy*

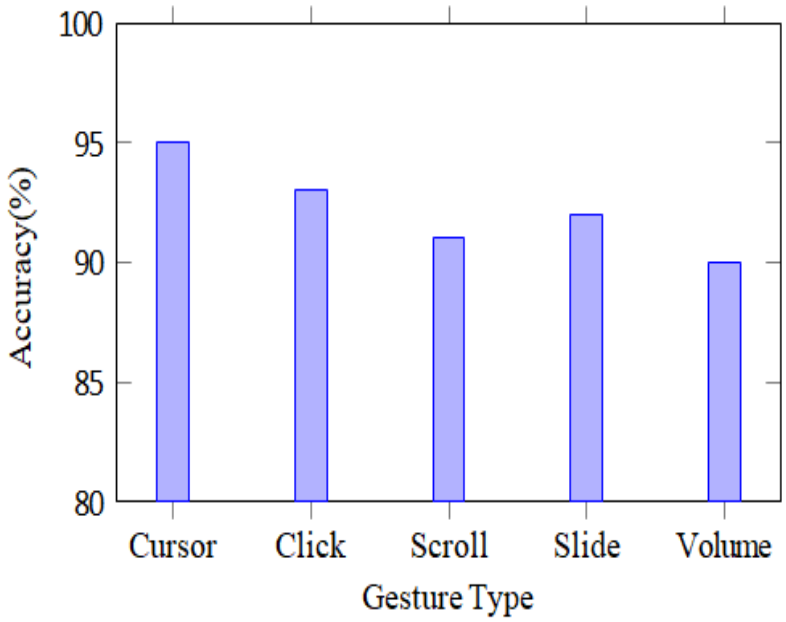


Fig.2.GestureRecognitionAccuracy

The accuracy graph shows that cursor movement gestures achieved the highest accuracy, while volume control gestures exhibited slightly lower accuracy due to variations in finger spread.

B. Precision and Recall Comparison

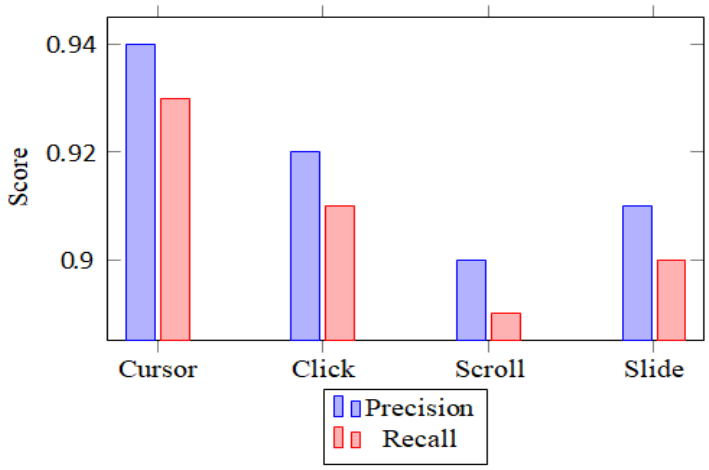


Fig.3.PrecisionandRecallComparison

Thisgraphcomparestheprecisionandrecallvaluesforeach gesture category. The results demonstrate that the proposed system maintains balanced classification performance across different gestures.

C. Latency Performance Analysis

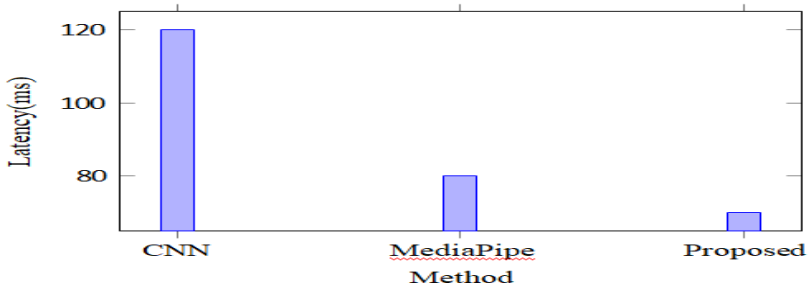


Fig.4.LatencyComparisonofGestureRecognitionMethods

The latency comparison shows that the proposed system achieves lower response time compared to traditional CNN- based gesture recognition systems.

D. GestureDatasetDistribution

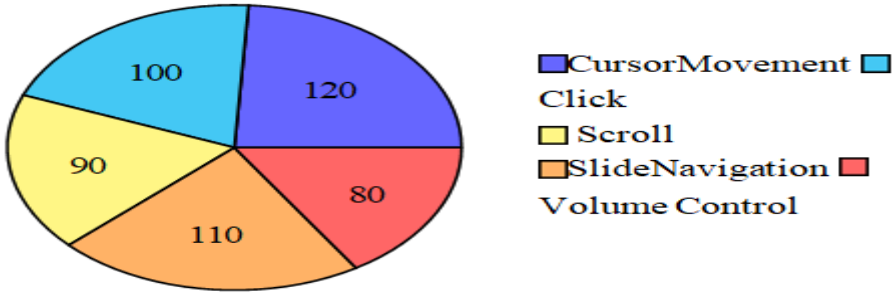


Fig.5.DistributionofGestureSamplesintheDataset

The pie chart illustrates the distribution of gesture samples used for evaluation. Cursor movement and slide navigation gestures form the largest portion of the dataset.

XIV. APPLICATIONS

Gesture-based interaction systems have numerous appli- cations across multiple domains where touchless interaction improves usability and safety.

1) Smart Classrooms: Gesture recognition systems can be usedtocontrolpresentationslides,annotatedigitalboards,and interact with educational content without requiring physical devices. This enables teachers to conduct interactive lectures while moving freely within the classroom.
2) Healthcare Environments: In medical environments such as operating rooms, touchless interfaces allow surgeons to interact with medical imaging systems without touching key- boards or mice. This helps maintain sterile conditions and reduces the risk of contamination.
3) AssistiveTechnologies:Gesture-basedinterfacescanassist individuals with physical disabilities by providing alternative interaction mechanisms for controlling computers and smart devices.
4) GamingandVirtualReality:Gesturerecognitioncan enhancegamingexperiencesbyenablingplayerstocontrol game characters or actions through natural hand movements.
5) PublicInformationKiosks:Touchlessgestureinterfacescanbeusedinpublicinformationsystemssuchasairportkiosks,ticketmachines, anddigitaldisplayswhereminimizing physicalcontactisbeneficial.
6) Smart Home Automation: Hand gestures can also be used tocontrolsmarthomedevicessuchaslights,televisions, and multimedia systems, providing a convenient and intuitive control mechanism.

XV. LIMITATIONS

Although the proposed gesture recognition system demon- strateseffectivereal-timeperformance,severallimitationsstill exist.

One of the primary challenges is sensitivity to lighting con- ditions. Poor illumination or excessive shadows may reduce the accuracy of hand landmark detection.

Background complexity also affects system performance. Whenthebackgroundcontainsobjectswithcolorssimilar to skin tones, the gesture recognition algorithm may produce incorrect detections.

Another limitation is the dependency on a single camera viewpoint. If the user's hand moves outside the camera frame or is partially occluded, the system may fail to detect gestures accurately.

Additionally, the system currently supports a limited set of predefined gestures. Expanding the gesture vocabulary would require more complex classification algorithms and larger training datasets.

Finally, computational performance may vary dependingon the hardware configuration, particularly when running the system on devices with limited processing power.

XVI. FUTURE WORK

Future research can improve the gesture recognition system in several directions.

One possible improvement is the integration of deep learn- ing models such as Convolutional Neural Networks (CNNs)or Transformer-based architectures for more robust gesture classification.

Multi-hand gesture recognition could also be implemented to enable collaborative interaction scenarios where multiple users interact with the system simultaneously.

Another potential enhancement is the integration of depth sensors or stereo cameras to improve gesture detection accu- racy under varying lighting conditions.

Augmented Reality (AR) and Virtual Reality (VR) inte- gration can also expand the application scope of the systemby enabling immersive gesture-based interaction in virtual environments.

Future systems may also incorporate adaptive learning mechanisms where the system learns personalized gestures from users, improving usability and customization.

Additionally, mobile and embedded implementations could enable gesture-based control for smartphones, wearable de- vices, and IoT systems.

XVII. CONCLUSION

This paper presented a gesture-based touchless control system designed for real-time human–computer interaction. The proposed system utilizes MediaPipe hand tracking and OpenCV image processing to detect hand landmarks and interpret user gestures.

The system enables users to control cursor movement, per- form clicking operations, scroll documents, navigate presen- tation slides, and control multimedia functions using natural hand gestures captured by a webcam.

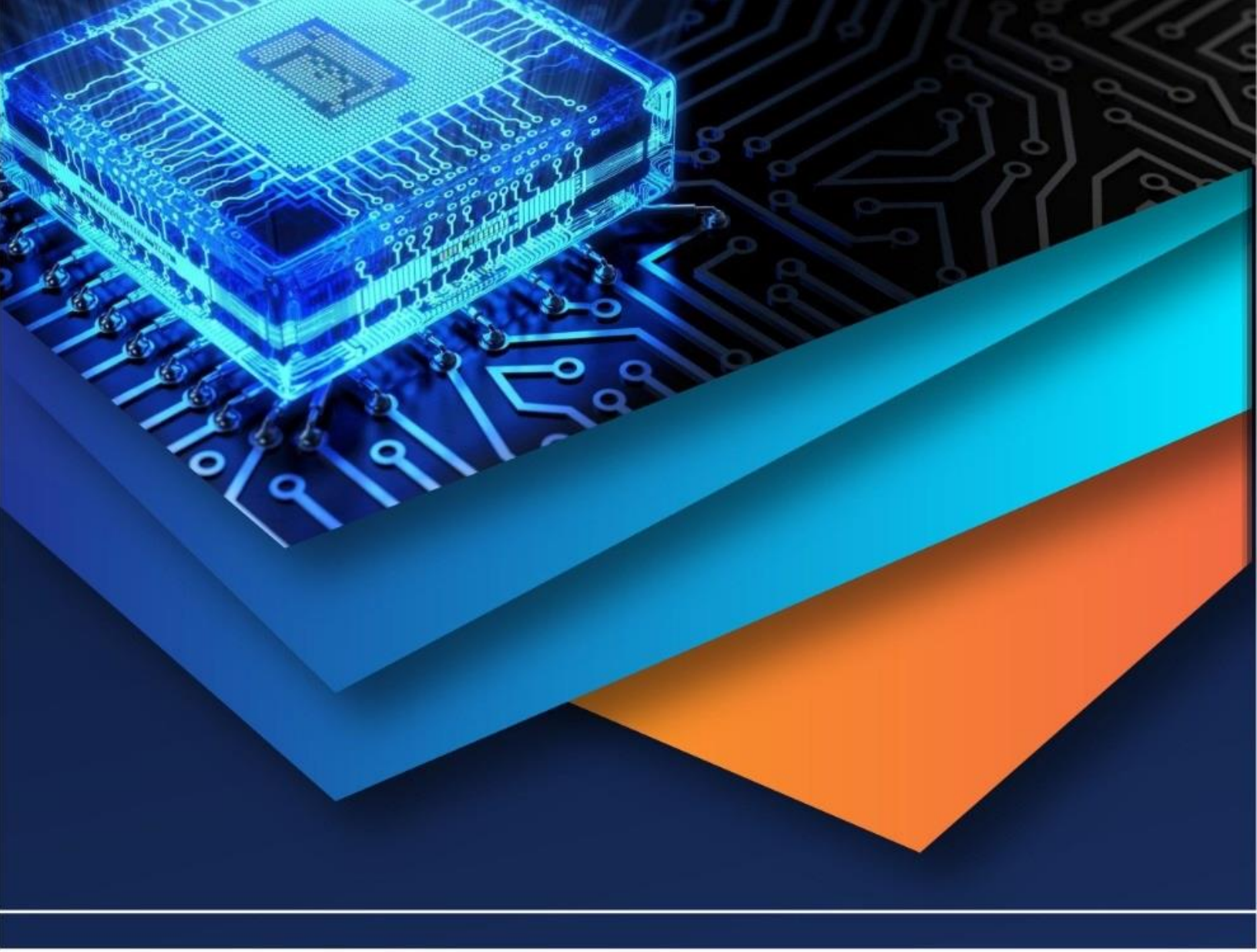
Experimental results demonstrate that the proposed ap- proach achieves high gesture recognition accuracy with an averageperformanceofapproximately92%whilemaintaining real-time processing speed of 25 frames per second.

Thesystemprovidesacost-effectiveandaccessiblesolution for touchless interaction without requiring specialized hard- ware. This makes it suitable for applications in education, healthcare, assistive technologies, and public interactive sys- tems.

Overall, the proposed gesture recognition framework con- tributes to the development of intuitive and hygienic interac- tion mechanisms for next-generation computing systems.

REFERENCES

[1] G. Bradski and A. Kaehler, Learning OpenCV: Computer Vision withthe OpenCV Library. O'Reilly Media, 2008.
[2] R. Szeliski, Computer Vision: Algorithms and Applications. Springer,2011.
[3] Google Research, "MediaPipe: A Framework for Building PerceptionPipelines," 2020. [Online]. Available: https://mediapipe.dev
[4] A. Haria, A. Subramanian, N. Asokkumar, S. Poddar, and J. S. Nayak,"HandGestureRecognitionforHumanComputerInteraction,"ProcediaComputer Science, vol. 115, pp. 367–374, 2017.
[5] H. Khanum and P. H. B, "Smart Presentation Control by Hand GesturesUsingComputerVisionandMediaPipe,"InternationalResearchJournalof Engineering and Technology, vol. 9, no. 7, pp. 1234–1238, 2022.
[6] S. Mitra and T. Acharya, "Gesture Recognition: A Survey," IEEETransactions on Systems, Man, and Cybernetics, vol. 37, no. 3, pp.311–324, 2007.
[7] T. Starner and A. Pentland, "Real-Time American Sign LanguageRecognitionfromVideoUsingHiddenMarkovModels,"inProceedingsof the IEEE International Symposium on Computer Vision, 1995.
[8] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust Hand GestureRecognitionwithKinectSensor,"inProceedingsoftheACMMultimediaConference, 2011.
[9] J.Shottonetal.,"Real-TimeHumanPoseRecognitioninPartsfroma Single Depth Image," in Proceedings of the IEEE Conference onComputer Vision and Pattern Recognition, 2011.
[10] I.Goodfellow,Y.Bengio,andA.Courville,DeepLearning.MITPress,2016.
[11] Y.LeCun,Y.Bengio,andG.Hinton,"DeepLearning,"Nature,vol.521,no. 7553, pp. 436–444, 2015.
[12] P. Viola and M. Jones, "Rapid Object Detection Using a BoostedCascade of Simple Features," in Proceedings of the IEEE Conferenceon Computer Vision and Pattern Recognition, 2001.
[13] W. Freeman and M. Roth, "Orientation Histograms for Hand GestureRecognition,"inInternationalWorkshoponAutomaticFaceandGestureRecognition, 1995.
[14] Z. Zhang, "A Flexible New Technique for Camera Calibration," IEEETransactions on Pattern Analysis and Machine Intelligence, vol. 22, no.11, pp. 1330–1334, 2000.
[15] J. Malik et al., "Object Recognition in Computer Vision," IEEE Trans-actionsonPatternAnalysisandMachineIntelligence,vol.33,no.8, pp.1545–1560,2011.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY