International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 05 | May - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

AI Virtual Mouse System Using Computer Vision to avoid COVID-19 spread

Dr. Santhosh Kumar S[*1], K.S Sharan Kumar[*2], Adinath S Jain[*3], Akash N Patel[*4]

[*1,2,3,4]Department of Information Science and Technology, Don Bosco Institute of Technology, Bengaluru, Karnataka, India.

.

ABSTRACT

In the contemporary digital landscape, enhancing human-computer interaction efficiency and intuitiveness is essential. Traditional input devices like mice and keyboards are being augmented by innovative approaches such as hand gesture recognition, which provides a more natural method of interaction. This paper aims to generate a virtual mouse controlled by hand gestures using computer vision and deep learning techniques. The system employs a webcam to capture live video of the user's hand movements. These movements are analyzed using convolutional neural networks (CNNs) to identify specific gestures, which are then translated into mouse operations like cursor movement, clicking, and scrolling. This solution is hardware-independent, utilizing only the device's camera, making it accessible and straightforward to use. The goal is to create a seamless and efficient interaction method, allowing users to control their computers with simple hand gestures from a distance.

Keywords: Convolutional Neural Network, Deep Learning, Hand Gesture Recognition, Virtual Mouse, Computer Vision, OpenCV

I. INTRODUCTION

With the rapid advancement of technology, the way humans interact with computers has evolved significantly. One area that has seen considerable innovation is Human-Computer Interaction (HCI), which focuses on creating more intuitive and efficient ways for participants to engage with computer systems. Traditional input devices such as keyboards and mice are being complemented and, in specific situations cases, replaced by more natural and intuitive methods. Gesture recognition has become a notable field within HCI, leveraging the natural human ability to communicate through movements with computers. Gesture recognition technology allows computers to interpret human gestures as commands, providing a more seamless and instinctive user experience. This initiative directs to develop a virtual mouse controlled by hand gestures using computer vision and deep learning techniques. By using a webcam to capture real-time video of hand movements, the system processes these movements through convolutional neural networks (CNNs) to identify specific gestures, which are then mapped to standard mouse functions such as moving the cursor, clicking, and scrolling.

This approach not only makes computer interaction more natural but also reduces the need for physical input devices, enhancing accessibility and convenience. The proposed system is hardware-independent, relying solely on the built-in camera of the device, making it easily deployable and user-friendly. This project promises to revolutionize the way users interact with their computers, providing a more efficient and engaging experience through the use of hand gestures.

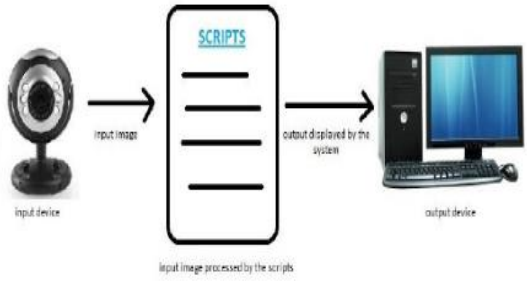


Figure 1: General Layout of Gesture Control

The above figure, illustrates the general layout of a virtual mouse system. The setup involves a webcam capturing hand gestures, which are processed by scripts on a computer. The processed gestures are then translated into cursor movements or clicks, which are displayed on the output device. This system enables users to control their computer using hand gestures, providing an alternative to traditional input devices like a physical mouse.

II. LITERATURE SURVEY

Yashas J and Shivakumar G [1] reviewed various approaches to Hand Gesture Recognition, emphasizing the differences between sensor-based systems and camera-based systems for data acquisition. Sensor-based systems provide direct hand interaction but are limited by physical constraints, while camera-based systems offer greater flexibility but introduce challenges such as background noise and lighting variability. Advanced methods like Adaptive Convolutional Neural Networks (ACNNs) using data augmentation were highlighted for their ability to improve classification accuracy by overcoming overfitting issues.

Sharma P and Sharma N [2] developed a strategy for detecting hand signals gestures using feature extraction techniques such as Principal Component Analysis (PCA) and Singular Value Decomposition (SVD). These elements were subsequentially categorized using Support Vector Machines (SVMs). However, this method faced challenges

with non-uniform backgrounds and the ability to recognize a diverse set of gestures.

Hakim NL et al., [3] investigated dynamic hand gesture recognition utilizing a 3D Convolutional Neural Network (3D CNN). By integrating data from RGB and depth cameras, they addressed the need for spatial-temporal features through a amalgamation of 3DCNN and Long Short-Term Memory (LSTM) models. The utilization of Finite State Machine (FSM) techniques further enhanced the accuracy of gesture classification by narrowing down the search space.

Cardenas EJ and Chavez GC [4] proposed a method combining temporal and spatial information from CNN descriptors and cumulative magnitude histograms for dynamic gesture recognition. Human poses were detected and key features were extracted, which were then classified using SVM models, demonstrating efficient performance.

Adithya V and Rajesh R [5] introduced an automatic hand posture recognition system using Convolutional Neural Networks (CNNs). This approach automated the feature extraction process, significantly reducing computational complexity while achieving high accuracy rates.

Munir Oudah et al., [6] examined various hand gesture recognition techniques, spanning from traditional computer vision methods to modern deep learning models. They discussed the advantages and limitations of each approach, highlighting the complex nature of gesture recognition tasks.

Adam Ahmed Qaid MOHAMMED et al., [7] explored hand detection using a specialized detector integrated with CNN architectures for gesture recognition. Despite challenges in cluttered environments, this approach outperformed traditional methods, demonstrating the potential of CNNs in handling complex backgrounds.

Raimundo F. Pinto Jr. et al.,[8] focused on gesture recognition using neural networks, emphasizing the importance of preprocessing techniques to prepare input data for CNN architectures. Their experiments showed that careful preprocessing could lead to significant improvements in recognition accuracy.

These studies collectively illustrate the advancements and ongoing challenges in the domain of hand gesture recognition, underscoring the critical role of CNNs and innovative methodologies in driving progress in this domain.

III. PROPOSED METHODS

The proposed gesture detection system for virtual mouse control is segmented into two primary stages. In the first stage, hand gestures are detected. In the second stage, these gestures are mapped to corresponding mouse actions. Gesture recognition is implemented using computer vision and deep learning techniques with a custom-built dataset. A new dataset has been created that contains seven distinct gestures, as shown in Figure 2. The raw images from the dataset are then processed into black and white images, as shown in Figure 3, to enhance recognition accuracy.

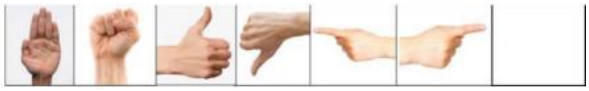


Figure 2: Raw data collected for the dataset

Figure 3: Gesture Data after Preprocessing

The CNN model in this project is trained by inputting image frames into three convolutional layers, each with a ReLU activation function. Max pooling layers are included to perform pooling operations, which are then flattened and passed through dense layers, as depicted in Figure 3.

In the proposed system, gestures are recognized by the CNN model. Images are preprocessed and resized before being fed into the model. The trained CNN model predicts one of the seven predefined gestures. Each gesture is mapped to a specific virtual mouse action, allowing the system to control the mouse based on the detected gesture.

A. Proposed Workflow

The system is organized into five modules. Figure 4 illustrates an overview of the system workflow, showing the gestures and their corresponding virtual mouse controls.

Figure 4: System Design Workflow

1) Capturing and Pre-processing Images

As the user performs hand gestures in front of the webcam, image frames are captured from the live video feed using OpenCV. These images are then converted to black and white, as illustrated in Figure 5, to enhance the precision of gesture prediction. The processed images are stored in their respective directories for further use.

When the webcam is active, the system displays two frames simultaneously on the screen, allowing users to capture images frame by frame using the read function. A mirrored image is provided for user convenience. Users are prompted to position their hand within the Region Of

Interest (ROI), which is delineated by a bounding box, to execute gestures effectively. Frames derived from the ROI are resized to dimensions of 120x120x1. The system indicates the count of images in each directory, incremented by pressing keys 0 to 6 on the keyboard to capture images. Captured images are then stored in their respective class directories.

During the capture process, preprocessed images are displayed in a small frame and concurrently saved to the dataset. To terminate the data collection process, users can simply press the escape key on the keyboard.

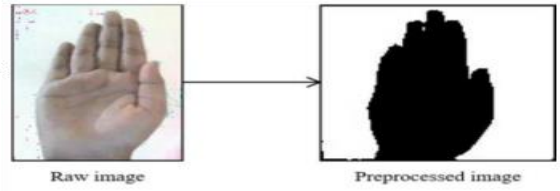
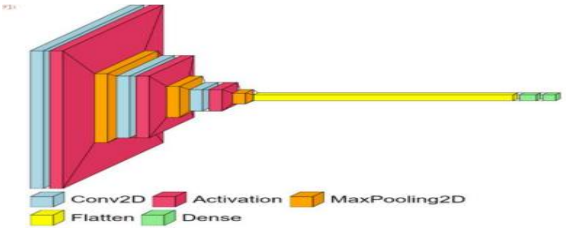


Figure 5: Data Preprocessing

2) Feature Extraction

To construct the convolutional neural network (CNN), Keras models and the necessary layers are imported. The CNN model begins with an input layer, followed by two convolutional layers. Each convolutional layer is accompanied by a ReLU activation function and a MaxPooling layer to reduce dimensionality. After these layers, a flattening layer is added to convert the 2D matrix data into a vector. This is followed by two fully connected layers: the first uses a ReLU activation function, and the second uses a SoftMax activation function for classification purposes. The architecture of the CNN model used for feature extraction and gesture classification is depicted in Figure. 6.

Figure 6: The proposed CNN model



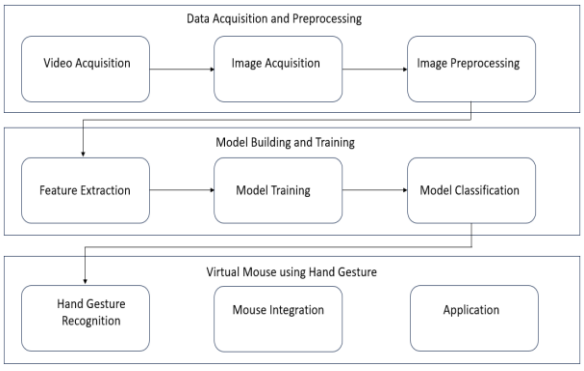


Figure 7: Proposed architecture diagram.

3) Model Training

The ImageDataGenerator class is utilized to generate batches of images for training and validation. The fit function is employed to train the model over a fixed number of epochs. Upon completion of the training process, the trained model is saved in JSON format, and the weights are saved separately using the save weights function.

4) Hand Gesture-Based Mouse Control

The trained model and its weights are loaded to enable hand gesture prediction. The PyAutoGUI library is employed for integrating hand gestures with virtual mouse controls. The user interface is created leveraging a web framework. When the user clicks the start button on the demo page, the webcam activates. The user performs hand gestures within the Region Of Interest (ROI), and the trained CNN model predicts the gestures. Each gesture is linked to a virtual mouse action through PyAutoGUI, using conditional statements to execute the appropriate control.

Each gesture is associated with a specific virtual mouse action and a label. The action is triggered once each time a gesture is detected. Users can exit the system by pressing the escape key. The video frame displays the predicted gesture and the corresponding action performed, allowing users to control the virtual mouse effectively.

B. System Architecture

The layout of a gesture recognition system typically involves three key components:

1) Data Acquisition and Pre-processing
2) Feature Extraction
3) Classification or Decision-making

The proposed system architecture is illustrated in Figure 7. It encompasses the three essential aspects needed for a gesture recognition system:

1) Data Acquisition and Preprocessing Phase

Image data is captured from a webcam in the form of a video. The video is broken into individual frames, which are converted to black and white images using OpenCV. These images are then stored in specific directories for further processing.

2) Model Building and Feature Extraction

A CNN model is constructed using Keras libraries. The ImageDataGenerator class is employed to preprocess the images, extracting only the necessary features. The design is trained using images from the training dataset.

3) Classification and Prediction

The design is compiled, and its accuracy is evaluated using a test dataset. Keras libraries are used to save and load the model, and to classify gestures into specific classes.

4) Integrating the Mouse Controls

Each gesture is integrated with corresponding control functions using the PyAutoGUI library. The predicted gestures are linked to execute specific actions

IV. RESULTS AND DISCUSSION

To analyze the proposed system in a real-world application, seven gestures were selected based on their practicality for the intended applications. Several criteria were subjected to test the model's robustness. The custom dataset consisted of 150 images per class for seven different classes, resulting in a total of 1050 images. These images were collected under proper lighting conditions with a static background and no background noise, as illustrated in Figure 8.

During real-time testing of the projected system with different sets of people, the system achieved good results under proper lighting conditions with no background noise. The system was evaluated using hand gestures from various individuals across three different CNN models, as detailed in Table III. By varying the count of convolutional and pooling layers, three unique architectures were created to evaluate their effectiveness and determine the optimal model according to their accuracy.

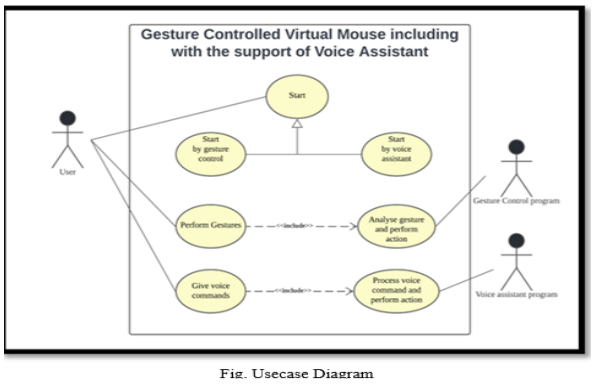


Fig. Usecase Diagram

Figure 8: Use Case Diagram of Gesture Control

The training time and testing time obtained from each CNN model are exhibited in Table I.

Table II. Comparison of accuracy with previous works

Author	Method	Dataset	Accuracy
Pinto RF et. al., [17]	Deep Learning with CNN 4	Custom	96.83%
Mohammed AA et.al., [16]	Deep RetinaNet based	Oxford	72.1%
		5-Signers	97.9%
		Egohands	93.1%
		Indian Classical	85.5%
Sharma P, Sharma N [11]	Feed-Forward Neural Network with PCA	Common Guesture	95.9%
		Common Posture	90.3%
	and SVD		
Hakim NL et. al., [12]	3DCNN+LSTM	Custom	95.8%
Proposed Model	Deep Learning with CNN	Custom	97.78%

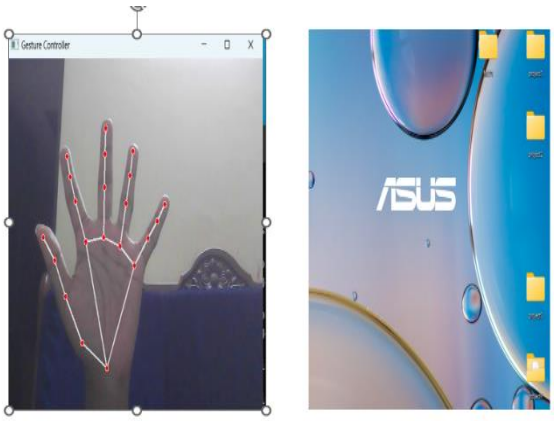


Figure 9: Data Collection for CNN Model

TABLE I. Training and Testing time

Time Taken	Training (in milliseconds)	Testing (in seconds)
CNN Model 3	460	0.13673686
CNN Model 2	694	0.05533218
CNN Model 1	828	0.15214204

The precision of the proposed model is examined with different models as exhibited in Table I

The real-time operation of the system when various hand gestures are exhibited in front of the webcam is illustrated from Figure 9 to Figure 12. These images demonstrate the working of the virtual mouse being controlled by specific hand gestures. For example, when a fist is shown to the webcam for the first time, the system activates the left click function. Showing the fist gesture again deactivates the left click. Similarly, the virtual mouse responds to six other

distinct gestures, each corresponding to a different control function.

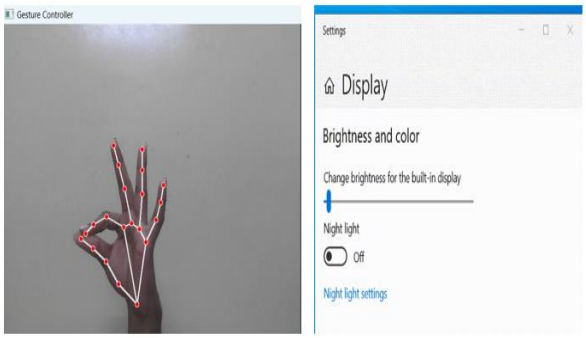


Figure 10: Brightness changed with Pinch Gesture

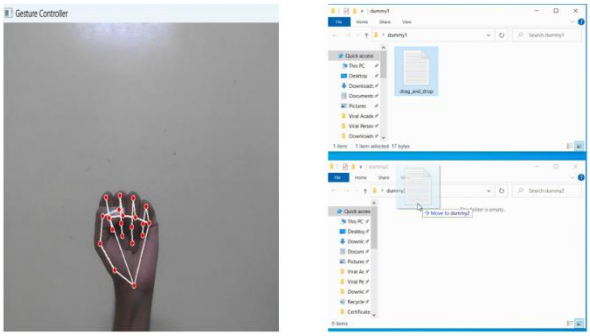


Figure 11: Drag mouse function with Fist Gesture

Figure 12: Left click with Bend Index Finger

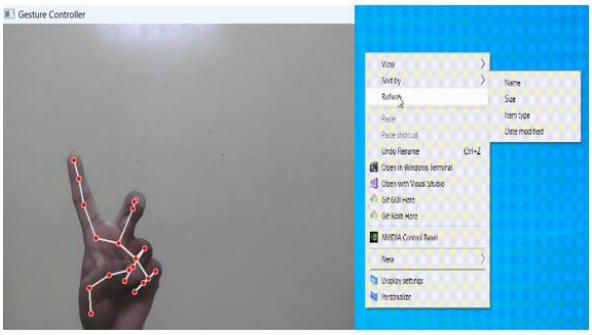


Figure 13: Right click with Bend Middle Finger

V. CONCLUSION

This project introduces a hand gesture recognition system for controlling a virtual mouse. Leveraging OpenCV for image capture, a 2-Dimensional Convolutional Neural Network for gesture feature extraction and prediction, and PyAutoGUI for mouse control integration, the system achieves a remarkable 98% accuracy rate with a custom dataset of 7 gestures. This real-time model offers users a seamless and cost-effective means of interacting with computer systems, exhibiting minimal latency. Moving forward, enhancements in gesture recognition across varying environmental conditions and the inclusion of additional hand gestures for diverse applications like word processing and web browsing are anticipated. Overall, this project signifies a significant advancement in real-time gesture-controlled technology for virtual mouse interaction.

ACKNOWLEDGEMENTS (optional)

The authors can acknowledge professor, friend or family member who help in research work in this section.

VI. REFERENCES

[1] Yashas J, Shivakumar G. Hand Gesture Recognition: A Survey. In2019 International Conference on Applied Machine Learning (ICAML) 2019 May 25 (pp. 3-8). IEEE.

[2] Sharma P, Sharma N. Gesture Recognition System. In2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU) 2019 Apr 18 (pp. 1-3). IEEE.

[3] Hakim NL, Shih TK, Kasthuri Arachchi SP, Aditya W, Chen YC, Lin CY. Dynamic hand gesture recognition using 3DCNN and LSTM with FSM context-aware model. Sensors. 2019 Jan;19(24):5429.

[4] Cardenas EJ, Chavez GC. Multimodal hand gesture recognition combining temporal and pose information based on CNN descriptors and histogram of cumulative magnitudes. Journal of Visual Communication and Image Representation. 2020 Aug 1; 71:102772.

[5] Adithya V, Rajesh R. A deep convolutional neural network approach for static hand gesture recognition. Procedia Computer Science. 2020 Jan 1; 171:2353-61.

[6] Oudah M, Al-Naji A, Chahl J. Hand gesture recognition based on computer vision: a review of techniques. journal of Imaging. 2020 Aug;6(8):73.

[7] Mohammed AA, Lv J, Islam MD. A deep learning-based Endto-End composite system for hand detection and gesture recognition. Sensors. 2019 Jan;19(23):5282

[8] Pinto RF, Borges CD, Almeida A, Paula IC. Static hand gesture recognition based on convolutional neural networks. Journal of Electrical and Computer Engineering. 2019 Oct 10;2019

[9] Cao Z, Hidalgo G, Simon T, Wei SE, Sheikh Y. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. IEEE transactions on pattern analysis and machine intelligence. 2019 Jul 17;43(1):172-86.

[10] Zhuang H, Yang M, Cui Z, Zheng Q. A method for static hand gesture recognition based on non-negative matrix factorization and compressive sensing. IAENG International Journal of Computer Science. 2017 Mar 1;44(1):52-9.

[11] Paul PK, Kumar A, Ghosh M. Human-Computer Interaction and its Types: A Types. International Conference on Advancements in Computer Applications and Software Engineering (CASE 2012), At Chittorgarh, India. –2012 2012 Dec 21

[12] Ritupriya G.Andurkar, Human–computer interaction. International Research Journal of Engineering and Technology, (ISSN: 2395-0072 (P), 2395-0056 (O)). 2015; 2(6):744

[13] Shukor AZ, Miskon MF, Jamaluddin MH, Bin Ali F, Asyraf MF, Bin Bahar MB. 2015. A new data glove approach for Malaysian sign language detection. Procedia Computer Sci 76:60–67

[14] Almeida SG, Guimara es FG, Ramí rez JA. 2014. Feature extraction in Brazilian sign language recognition based on phonological structure and using RGB-D sensors. Expert System Appl 41(16):7259–7271

[15] Murakami K, Taguchi H. 1991. Gesture recognition using recurrent neural networks. In: Proceedings of the ACM SIGCHI conference on Human factors in computing systems, pp 237–242. https ://dl.acm.org/doi/pdf/10.1145/10884 4.10890

[16] Wang RY, Popovic J. 2009. Real-time hand-tracking with a color glove. ACM Trans Graph 28(3):63