# Interactive Human-Computer Systems via Hand Tracking and Gesture Recognition

**Ankit Siwatch**
**Department of Computer Science Engineering**
**Lovely Professional University, Phagwara**
**ORCID: 0009-0006-4722-9138**
**ankitsiwach3011@gmail.com**

**Virat**
**Department of Computer Science Engineering**
**Lovely Professional University, Phagwara**
**ORCID: 0009-0000-7917-9354**
**viratkumar9091@gmail.com**

**Nikhil Singh**
**Department of Computer Science Engineering**
**Lovely Professional University, Phagwara**
**ORCID: 0009-0003-3196-7644**
**nitins706888@gmail.com**

**Kaushal Pathak**
**Delivery and Student Success**
**upGrad Education Private Limited**
**Bangalore, Karnataka, India**
**ORCID: 0009-0002-9208-619X**
**pathak3982@gmail.com**

**Shamneesh Sharma**
**Delivery and Student Success**
**upGrad Education Private Limited**
**Banglore, Karnataka, India**
**ORCID:0000-0003-3102-0808**
**shamneesh.sharma@gmail.com**

***Abstract***

**Hand tracking and gesture recognition play a crucial role in human-computer interaction. While many solutions have been proposed, the problem remains unsolved due to the complexity of hand movements and challenges like self- occlusion. These issues become even more difficult in interactive applications that require real-time performance.(Yeo et al., 2015)With webcams now widely available, there is an opportunity to develop computer vision systems that enable new ways to interact with computers. In this work, we present a system that allows users to control the operating system's cursor without touching a mouse—simply by gesturing in the air. Our system uses OpenCV and the X windowing system to track the user's index finger and thumb via a webcam. The user can move their hand in the desired direction to control the cursor and perform mouse actions by pointing toward the camera and breaking a predefined interaction plane.This project serves as a proof of concept for a hands-free computer control system using standard webcams, opening the door for more intuitive and accessible ways to interact with digital devices.**

***Keywords:- Computer vision, Mediapipe, Human computer interaction, Gesture recognition.***

## I. Introduction

In a world where technology is becoming more intuitive, the way we interact with computers is evolving beyond keyboards and touchscreens. Hand gesture recognition is emerging as a powerful and natural way to bridge the gap between humans and machines, allowing users to communicate through simple movements instead of traditional input devices. Whether it's controlling a smart home, navigating virtual reality, or enhancing accessibility for individuals with disabilities, gesture-based interfaces have the potential to make digital interactions more seamless and immersive.(Chang et al., 2023a)

This paper explores the advancements in hand gesture recognition, focusing on its role in Human-Computer Interaction (HCI). We discuss the various techniques used to detect and interpret gestures, including computer vision-based methods and sensor-driven approaches. We also explore the key hurdles in gesture recognition systems, including precision, real-time responsiveness, and the influence of external environmental conditions (Chang et al., 2023b). Gaining insight into these elements paves the way for developing more intuitive and effective user interfaces, moving us closer to seamless, natural interaction with digital technologies.[1]

## II. Literature survey

Hand Gesture Recognition (HGR) involves enabling machines to interpret the motions and shapes formed by human hands—essentially teaching computers to "understand" gestures the way humans naturally do. This field is an intersection of multiple technologies, including artificial intelligence, computer vision, and image processing. The primary objective is to allow systems to accurately detect and interpret hand gestures. The recognition process typically follows a series of stages. It begins with collecting visual data—images or video sequences that capture various hand gestures. After gathering this data, the visuals are refined to enhance clarity, ensuring that the system can easily detect crucial details. This enhancement phase prepares the images for analysis. Next comes segmentation, where the system isolates the hand region from the rest of the image, eliminating unnecessary background information. Once the hand has been clearly identified, the system moves on to feature extraction. This step involves analyzing specific characteristics—like the shape, motion, or orientation of the hand—that are essential for identifying the intended gesture.[2]

Finally, after collecting all the important features, the last step is classification. In this stage, the computer tries to match the observed gesture to a known category—like a peace sign, a thumbs-up, or a stop sign—by comparing it with the gestures it has already learned during training. When it comes to gestures, there are two main types: static and dynamic. Static gestures are simple hand poses that don't move. These can be understood from just one image, like showing an open palm or a closed fist. Dynamic gestures, on the other hand, involve movement over time. A good example would be waving goodbye or gestures used in sign language. These need a series of images or a video so that the computer can understand the movement.

One of the most important parts of any gesture recognition system is how it collects the data. Different systems use different ways to do this. Vision-based systems use regular cameras to capture hand gestures, just like taking a photo or recording a video. These systems focus on what the camera can "see" and use that information to detect and interpret hand movements. Other systems might use wearable devices or sensors that can feel the hand's movement and direction directly. For example, a glove with sensors can track finger movements in real time.[3]

However, this discussion is mostly about vision-based systems that use images and videos to recognize static hand gestures. In vision-based systems, the camera captures the hand image, then computer programs help to clean up the image, focus only on the hand, and pick out the useful parts of the image. After that, special machine learning or deep learning models help the system learn what each gesture means. These models are designed to work like the human brain, allowing the system to learn from examples and make decisions based on what it has seen before. To wrap it up, hand gesture recognition is a clever mix of technology that helps machines understand hand signs. From collecting data to interpreting gestures, each step is important.

Whether the gestures are still or moving, vision-based systems play a big role in teaching computers how to "see" and "understand" hands, just like we do in everyday life.

Hand gesture recognition is steadily becoming a valuable part of our everyday technology, with its presence expanding across a variety of industries. In gaming and virtual reality, for instance, users can control in-game actions or interact with digital environments using only their hands—eliminating the need for traditional controllers and creating a more immersive, lifelike experience.In healthcare settings, particularly in surgical environments, gesture-based systems allow medical professionals to access patient records or control equipment without direct contact, which helps maintain sterile conditions. Within smart homes, simple hand motions can be used to operate devices like lights, music systems, or thermostats. This is especially beneficial for individuals with mobility limitations who may struggle with conventional controls.

The automotive sector is also integrating gesture control, enabling drivers to adjust settings like audio volume or GPS navigation through hand movements—reducing the need for physical interaction and enhancing safety by helping drivers stay focused. In robotics, machines can interpret hand signals as commands, improving human-robot collaboration in settings such as manufacturing plants or during emergency rescue operations.[4]

Education is another area seeing the benefits of gesture recognition, where it's being used to foster more dynamic and engaging learning experiences. As this technology advances, it continues to become faster and more reliable—even under less-than-ideal conditions like dim lighting or visually complex backgrounds. Researchers are also focused on improving system adaptability so that it can accurately interpret gestures from users with varying hand shapes, skin tones, and movement patterns.In essence, hand gesture recognition has moved well beyond the experimental stage and is already embedded in practical, real-world applications. With ongoing innovation, it promises to make our interactions with technology more seamless, intuitive, and inclusive for everyone

## III. Methodology

Hand gesture recognition has become a major focus within the field of Human-Computer Interaction (HCI), with ongoing research aimed at enhancing both its accuracy and efficiency. In its early stages, the technology heavily depended on hardware-based solutions—such as sensor-equipped gloves—which, while offering precise tracking of hand movements, were often impractical due to their high cost, bulkiness, and lack of user-friendliness for everyday scenarios. This limitation led to a shift toward vision-based techniques, particularly those driven by advancements in computer vision and deep learning. These approaches enable users to interact with systems without physical contact, offering a more intuitive and seamless experience [5]Fig1.A key player in the development

of gesture recognition technologies is OpenCV, a widely-used open-source library for computer vision tasks. Researchers have extensively utilized its capabilities for processing visual data—employing methods such as skin tone detection, contour identification, and feature extraction to recognize and categorize different hand gestures. Techniques like background subtraction and image thresholding help isolate the hand from its surroundings, forming the basis for gesture analysis. To improve precision, many systems now incorporate machine learning and deep learning techniques into their pipelines [6]

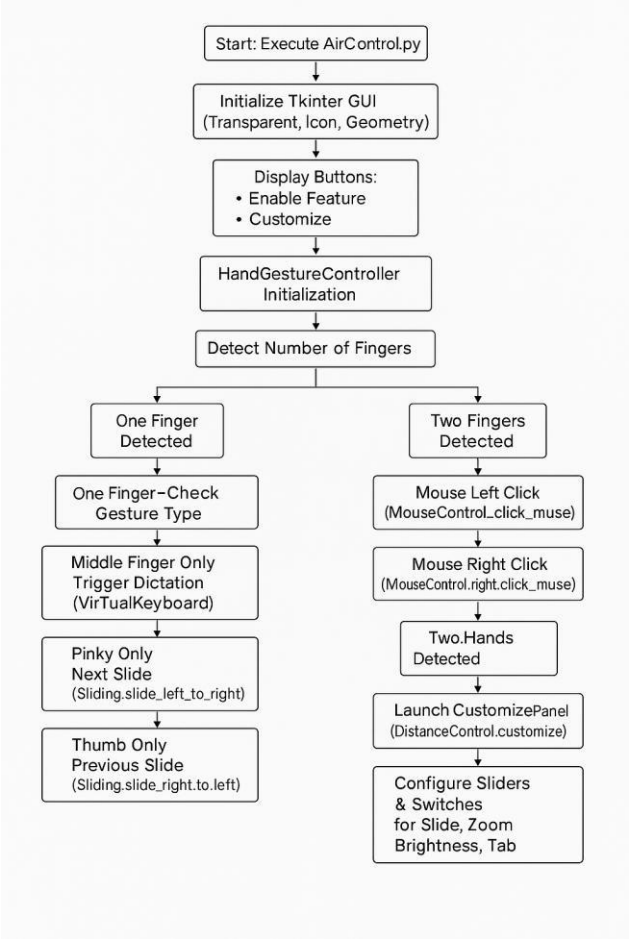


Fig 1. Flow Chart

For more advanced applications, researchers often combine OpenCV with deep learning platforms like PyTorch and TensorFlow. Convolutional Neural Networks (CNNs) have proven particularly effective in identifying both static hand positions and dynamic movements, while Long Short-Term Memory (LSTM) networks enhance the system's ability to interpret gesture sequences in real-time[7]. Tools such as Haar cascades, along with more modern solutions like MediaPipe, have further boosted the performance of real-time hand tracking, resulting in faster and more dependable gesture-based systems.Despite these advancements, challenges still exist. Factors like varying lighting conditions, diverse skin tones, and cluttered backgrounds can impact the accuracy of[8]

recognition systems. That's why recent research is focusing on hybrid models that combine OpenCV's efficient image processing with deep learning's adaptability. The ultimate goal is to develop more accurate, scalable, and user-friendly solutions that bring us closer to effortless, gesture-based interactions with technology.

### *A. Proposed Method*

For the proposed system, a simple and efficient algorithm capable of working in real-time and with a small computational effort is proposed. The system pipeline comprises four main steps: Frame Recording, Hand Recognition, Hand Segmentation, and Gesture Recognition. Specifically, for each image captured by the camera, a hand detection process is performed to identify the portions of the image where hands are present. Subsequently, a hand segmentation step is conducted to generate a mask that represents the shape of the detected hands. The resulting mask is used as input for the Gesture Recognition step, which predicts the executed gesture.[9]

### *B. Hand and Finger Detection*

The hand detection process works by identifying and isolating hand pixels in an RGB image while also pinpointing the center of each detected hand. To do this, the system combines two different masking techniques: color analysis in the HSV color space and foreground detection. The color-based approach uses predefined skin tone thresholds in the HSV domain, which can be adjusted based on lighting conditions or individual skin variations. The Hue range is set between 6 and 28, while Saturation and Value can vary from 0 to 255 [10]Meanwhile, foreground detection helps separate moving objects from static ones by analyzing frame-to-frame differences, allowing the system to distinguish hands from the background.

Once the mask is created, it is applied to the original camera feed to generate the Hand Pixel Mask, highlighting the detected hand regions. To accurately determine hand positions, the system uses k-means clustering on this mask. However, the number of clusters (k) must be carefully chosen for accurate results. Instead of setting k manually, the system relies on the Elbow Algorithm, which automatically finds the optimal k by minimizing the Sum of Squared Distances (SSD)—a measure of how well the detected pixels fit their assigned clusters. Since pixel density changes depending on the hand's distance from the camera, the SSD is normalized to maintain consistency. The Elbow Algorithm stops increasing k when further improvements become minimal, using a slope threshold to identify the point where the function starts to level off. At this point, the last recorded k value is selected.[11]

To further improve accuracy, the system minimizes false-positive clusters by tracking hand movements across frames. It does this by comparing centroids from the current frame with those from previous frames using Euclidean distance. Since k is recalculated in each frame, some centroids may be missed, especially if hands are too close together or overlapping. To make tracking more reliable, the system dynamically adjusts centroid positions, ensuring

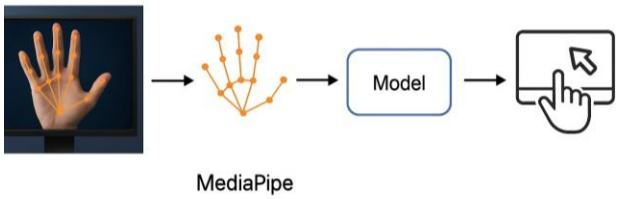


Fig 2. **MediaPipe**

smooth hand tracking even if occasional detection errors occur. As long as the system processes enough frames per second, these errors won't significantly impact performance. A great deal of effort has been put into accurately computing hand centroids, as they play a crucial role in filtering out unwanted artifacts—like mistakenly identifying other skin regions as hands—ensuring a robust and efficient hand detection system.

### *C. Background Subtraction*

The first step in the tracking system is to separate hand pixels from the background. To reduce noise, all captured images are first processed with a 5×5 Gaussian filter and then scaled down by half in each dimension. This filtering helps smooth the image and remove unnecessary pixel variations. Since the stereo cameras are fixed above a static workspace, a simple background subtraction method is used to distinguish the moving hand from the unchanged background.(Ezquerro et al., 2024) Once the foreground hand is isolated, further image processing techniques can be applied to enhance accuracy in gesture recognition and tracking.[12]

### *D. Region Extraction*

Once the skin regions are detected, the next step is to identify which ones correspond to the left and right hands. Since background subtraction and skin segmentation may leave behind small noise regions, we assume that the hands will be the largest detected regions. To extract them, we use binary image processing techniques and connected component analysis to identify and analyze contours. By focusing on the largest connected skin regions, the system can accurately isolate the hands while filtering out any unwanted noise, ensuring a clean and precise foundation for further gesture recognition and tracking.[13](Gore Karwankar, n.d.)

### *E. Feature Extraction*

To detect the fingertips of the thumb and index finger for each hand, we identify peaks along the hand's contour. First, we represent contour points as 3D vectors in the xy-plane and compute their cross product. If the z-component of the cross product is positive, the point is labeled as a peak; if negative, it is marked as a valley. To refine detection, non-maximal suppression is applied, ensuring that only the strongest peaks and valleys are selected. This process helps filter out unnecessary points and accurately locate fingertips, improving the reliability of gesture recognition.

### *A. One-Finger Gestures*

These are the easiest ones, just using a single finger for simple tasks:

- **Dictation Gesture (Middle Finger Only)** So, imagine you want to start dictating—like talking to your phone or a voice assistant. You just extend your middle finger, and *boom*, it's ready to listen to whatever you say. It's like giving your assistant a "Hey, listen to me!" sign.
- **Next Slide (Pinky Finger Only)** You know how you're giving a presentation, and you need to move to the next slide? Extend your pinky, and it's like flipping to the next page of your notes. Super simple!
- **Previous Slide (Thumb Finger Only)** Need to go back to the last slide? Just pop out your thumb. It's like hitting the "back" button, just with your hand. Easy, right?

### *B. Two-Finger Gestures*

Now we get a little more complicated, using two fingers for tasks that need more control:

- **Mouse Movement (Thumb and Index)** Picture this: you extend your thumb and index finger like you're pointing at something, and suddenly you can move the mouse on the screen! It's like guiding the cursor with your hand, super intuitive.
- **Volume Up (Index and Pinky)** Want to crank up the volume? Just stick out your index and pinky. It's like you're telling the system, "Turn it up!" and it listens. No need to fumble with volume buttons.
- **Volume Down (Middle and Pinky)** Similarly, if you need to turn the volume down,

just extend your middle and pinky. It's basically like saying, "Shh, a bit quieter please."

| Sr. no | Gesture | Fingers Used | Functionality |
|---|---|---|---|
| 1. | Dictation Gesture | Middle finger extended | Triggers dictation using the Virtual Keyboard |
| 2. | Mouse Movement | Thumb & Index extended | Moves mouse based on index finger position |
| 3. | Left Click | Thumb, Index & Pinky extended | Performs a left mouse click |

TABLE 1:- GESTURE FUNCTIONALITY

- **Arrow Up (Index and Middle)** Extend your index and middle fingers, and it's like hitting the "up" arrow key on your keyboard. This one's great for scrolling up through lists, pages, or any feed.
- **Arrow Down (Ring and Pinky)** Just like the "up" gesture, but instead, extend your ring and pinky fingers, and it scrolls the page down. It's like pressing the "down" arrow to move through content. Simple!

### C. *Three-Finger Gestures*

Now we're using three fingers, so things get even more interactive:

- **Left Click (Thumb, Index, and Pinky)** you want to click on something, but there's no mouse in sight. Instead, you make a gesture by extending your thumb, index finger, and pinky. That simple hand sign acts just like a mouse click. It's a clever, hands-free way to select items and comes in especially useful when you're going for a more intuitive control method.
- **Right Click (Index, Ring, and Pinky)** Think of this as a right-click, but without the mouse. By stretching out your index, ring, and pinky fingers, you're triggering the same action—bringing up a context menu with extra options. It's a smooth, hands-free way to access more controls without ever needing to reach for a physical device.

### *D. Five-Finger Gestures*

Here, we're using all five fingers, and things get pretty smooth for more advanced control:[14]

- **Tab Switch Forward (Hand Moves Right)** Got a bunch of tabs open? Move your hand to the right with all five fingers extended, and you've just switched to the next tab. It's like flipping through tabs, but way cooler.
- **Tab Switch Backward (Hand Moves Left)** Need to go back to the previous tab? Move your hand to the left with all five fingers extended. It's like going back a step, but in the browser. Super easy!

### *E. Two-Handed Gesture*

This one's for zooming, and you'll need both hands:

- **Zoom In/Out (Both Hands, Index Fingers Extended)** You know how you pinch-to-zoom on your phone? Same idea here! Extend both your index fingers and move your hands closer or farther apart. The system measures the spacing between your fingers to control zoom functions—bringing the screen closer or pulling it back. It's similar to the familiar pinch-to-zoom gesture, but instead, it uses full-hand movements for a more dynamic, hands-free experience.[15]

## IV. Results and Discussion

To test how well our Hand Gesture Recognition System performs in the context of Human-Computer Interaction (HCI), we carried out multiple experiments under varying environmental settings and lighting conditions. These trials were aimed at measuring the system's real-time gesture detection capabilities, focusing on its accuracy, speed, and ability to remain reliable across different scenarios.

### *A. Experimental Setup*

The system was tested using a webcam with a resolution of 640x480 pixels, running at 30 frames per second (FPS). The testing environment included varying lighting conditions, such as natural daylight, indoor artificial lighting, and dimly lit surroundings, to measure the system's adaptability. The hardware setup consisted of a standard laptop with an Intel Core i7 processor and 16GB RAM, ensuring real-time processing.[16]

A total of 20 participants took part in the experiments, each performing a predefined set of gestures, including Start, Move, Stop, and No-Hand gestures, along with directional movements (Left, Right, Front, Back). Each participant performed the gestures 50 times, resulting in a dataset of 1,000 gestures for evaluation.

### *B. Hand Gesture Recognition*

Our system recognizes real-time hand gestures in unconstrained environments using a predefined set of gestures and directions. The four gestures include: an open hand with fingers spread, an open hand with fingers

together, a fist, and a no-hand gesture (when the hand is partially or fully out of view). These correspond to Start, Move, Stop, and No-Hand actions.(Zhang et al., 2024)[17]

During the Move gesture, users can perform directional movements: Left (wrist rotation left), Right (wrist rotation right), Front (hand moving closer to the camera), and Back (hand moving away). The system ensures smooth transitions between valid gestures for accurate interaction.(Yadav & Bhattacharya, 2016)

Given these limitations, we opted for the first method, as it offers better accuracy by directly mapping the finger's position to the screen, ensuring a more precise and responsive cursor movement.(Trigueiros et al., n.d.)

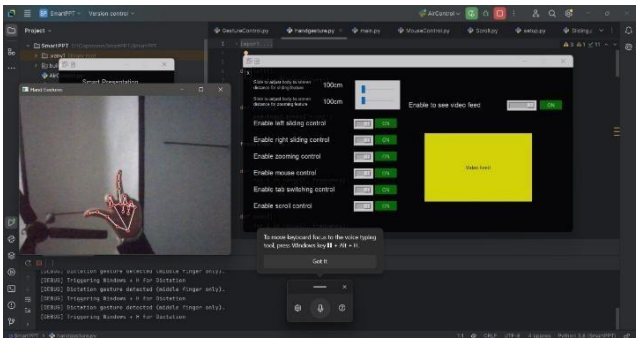

Fig 3a. **Gestures**

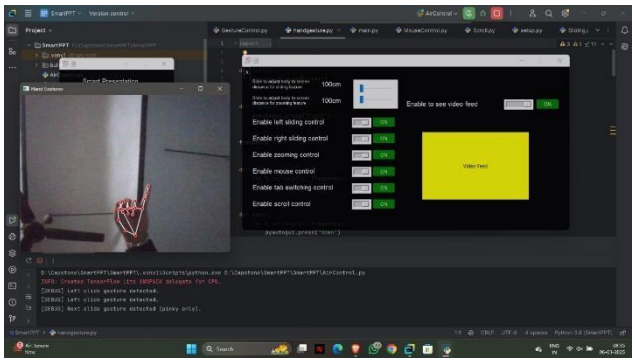

Fig 3b. **Gestures**

### *C .Hand Tracking*

Hand tracking ensures that once a hand is detected, it is continuously followed across frames for accurate gesture recognition. The system first detects the hand using color-based filtering in the HSV space and foreground detection.

For tracking, OpenCV offers techniques like contour detection and centroid tracking, where the hand's centroid is extracted and tracked across frames using Euclidean distance. To refine accuracy, k-means clustering filters out noise.(Molchanov et al., 2016)

Challenges like fast movements or occlusions are handled using Kalman filtering, which predicts the next hand position. A bounding box approach limits the search area, improving efficiency. Advanced frameworks like MediaPipe Hands enhance precision by detecting key hand landmarks.

For real-time performance, frame skipping and multi-threading optimize speed. By combining contour-based tracking, centroid comparison, and predictive modeling, the system ensures smooth, accurate tracking, making gesture recognition reliable and efficient for human-computer interaction.[18]

### *D. Accuracy and Recognition Rate*

The system's overall gesture recognition accuracy was 92.5% across all participants and conditions. However, we observed variations based on environmental factors:

- **Bright/Natural Light:** Achieved the highest accuracy of **96%**, as hand contours were well-defined.
- **Artificial Indoor Light:** Accuracy dropped slightly to **91%**, likely due to shadows and reflections.
- **Low Light Conditions:** Performance declined to **85%**, as hand segmentation became more challenging.

The system performed best when the hands were fully visible and distinct from the background. Overlapping hands or fast movements caused occasional misclassifications, particularly in low-light settings(Rohit et al., 2023).

### *E. Real-Time Responsiveness*

On a system processing at 30 FPS, the average response time for gesture recognition was 120 milliseconds, making interactions feel natural and lag-free. However, on lower-end machines (with slower CPUs), processing delays were noticeable, particularly when using more computationally intensive tracking methods.(Kumaran et al., 2021)

## *V.* Error Analysis and Challenges

While the system performed well overall, certain challenges were identified:

- **Hand Occlusion:** When hands overlapped or partially exited the camera frame, recognition errors increased.
- **Skin Color Variations:** The HSV-based segmentation occasionally misclassified non-hand regions, particularly in non-uniform lighting.

- **Fast Hand Movements:** Rapid gestures sometimes caused motion blur, leading to reduced accuracy.

## VI. Conclusion

Our experiments demonstrate that the system is highly effective in recognizing hand gestures for HCI, with real-time performance and high accuracy. Despite minor limitations, it offers a practical and intuitive way to interact with computers without physical input devices.(Munir et al., 2021)

## VII. Improvements and Future Work

To further enhance accuracy and robustness, integrating deep learning-based tracking (e.g., MediaPipe Hands) could provide more precise keypoint detection. Additionally,(Prasad et al., 2023) adaptive background subtraction could improve performance in varying lighting conditions.

## *VIII. References*


[1] H. S. Yeo, B. G. Lee, and H. Lim, "Hand tracking and gesture recognition system for human-computer interaction using low-cost hardware," *Multimed Tools Appl*, vol. 74, no. 8, pp. 2687–2715, Apr. 2015, doi: 10.1007/s11042-013-1501-1.

[2] V. Chang, R. O. Eniola, L. Golightly, and Q. A. Xu, "An Exploration into Human–Computer Interaction: Hand Gesture Recognition Management in a Challenging Environment," *SN Comput Sci*, vol. 4, no. 5, Sep. 2023, doi: 10.1007/s42979-023-01751-y.

[3] M. Oudah, A. Al-Naji, and J. Chahl, "Hand Gesture Recognition Based on Computer Vision: A Review of Techniques," Jul. 01, 2020, *MDPI*. doi: 10.3390/JIMAGING6080073.

[4] R. Azad, B. Azad, N. B. Khalifa, and S. Jamali, "Real-Time Human-Computer Interaction Based on Face and Hand Gesture Recognition," *International Journal in Foundations of Computer Science & Technology*, vol. 4, no. 4, pp. 37–48, Jul. 2014, doi: 10.5121/ijfcst.2014.4403.

[5] C. Manresa-Yee, J. Varona, R. Mas, and F. J. Perales, "Hand tracking and gesture recognition for human-computer interaction," in *Progress In Computer Vision And Image Analysis*, World Scientific Publishing Co., 2009, pp. 401–412. doi: 10.1142/9789812834461_0022.

[6] M.-A. Okkonen, V. Kellokumpu, M. Pietikäinen, and J. Heikkilä, "LNCS 4522 - A Visual System for Hand Gesture Recognition in Human-Computer Interaction." [Online]. Available: http://www.ee.oulu.fi/mvg

[7] P. Trigueiros, F. Ribeiro, and L. P. Reis, "Hand Gesture Recognition System based in Computer Vision and Machine Learning."

[8] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. K. I. Ieee, "Figure 2: Classification of dynamic gestures with R3DCNN. ["Online Detection and Classification of Dynamic Hand Gestures with Recurrent 3D Convolutional Neural Networks" by," 2016.

[9] M. V Gore Karwankar, "Human Computer Interaction using Hand Gesture Recognition." [Online]. Available: www.ijert.org

[10] R. Joshi, "Hand Gesture Detection for Human-Computer Interaction: A Natural Language Processing Approach." [Online]. Available: www.ijert.org

[11] K. Yadav and J. Bhattacharya, "Real-time hand gesture detection and recognition for human computer interaction," in *Advances in Intelligent Systems and Computing*, Springer Verlag, 2016, pp. 559–567. doi: 10.1007/978-3-319-23036-8_49.

[12] B. Kumari, P. Yadav, D. Singh Chauhan, H. Goel, and S. Dutt Sharma, "Hand Gesture Recognition," 2022. [Online]. Available: www.ijnrd.org

[13] S. S. Rohit, R. Kandoi, and S. Kumar, "Hand Gesture Recognition for Human Computer Interaction through KNN Algorithm and Mediapipe," *International Journal of Computer Sciences and Engineering*, vol. 11, no. 4, pp. 14–18, Apr. 2023, doi: 10.26438/ijcse/v11i4.1418.

[14] W. Zhang, J. Wang, and F. Lan, "Dynamic hand gesture recognition based on short-term sampling neural networks," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 1, pp. 110–120, Jan. 2021, doi: 10.1109/JAS.2020.1003465.

[15] N. Kumaran, M. Sri Anurag, M. Sampath, and A. Professor, "N Kumaran, Hand Gesture Recognition Using Transfer Learning Techniques," 2021.

[16] M. M. Prasad, S. Narayan, U. Scholar, A. Professor, and U. Scholars, "SIGN LANGUAGE AND HAND GESTURE RECOGNITION SYSTEM," 2023. [Online]. Available: www.ijcrt.org

[17] S. Munir, S. Mushtaq, A. Nadeem, and S. Zahra, "Hand Gesture Recognition: A Review," *LUME*, vol. 10, p. 5, 2021, [Online]. Available: www.ijstr.org

[18] S. Bakheet and A. Al-Hamadi, "Robust hand gesture recognition using multiple shape-oriented visual cues," *EURASIP J Image Video Process*, vol. 2021, no. 1, Dec. 2021, doi: 10.1186/s13640-021-00567-1.