International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 05 | May - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Real-Time Hand Gesture Recognition for Media Player using CNNs

Krishna Swaroop A, Sudarshan Shweth Nazare, Suhas S K, Suman S Y, Uma C M

Malnad College of Engineering,Hassan

Email: ksa@mcehassan.ac.in , sudhu102shweth@gmail.com , sksuhassk@gmail.com , sumansy18.17@gmail.com , umacm06@gmail.com

***Abstract*— A promising method for hand gesture identification is thermal imaging, which offers more robustness than conventional RGB camera-based systems in demanding settings. Thermal imaging provides resilience in a variety of settings, especially in different lighting circumstances, in contrast to RGB camera-based solutions. The study examines cutting-edge instruments and methods, such as real-time gesture mapping, CNN-based feature extraction, and effective model training approaches. The potential of thermal imaging to achieve accurate and user- friendly control over media functions including play, pause, and volume adjustments is highlighted by key discoveries. For smooth operation on devices with limited computational resources, the research focuses on resource-efficient architectures. This work lays the groundwork for creating gesture-controlled systems that are flexible, dependable, and easy to use.**

I. INTRODUCTION

A crucial area of study with revolutionary potential for human-computer interaction is hand gesture recognition. Although computer vision has been widely used in conventional methods for applications like human-robot collaboration, these methods struggle in dynamic and unstructured situations [1]. Enhancing recognition accuracy and robustness in a variety of circumstances has been achieved through the integration of digital vision into different modalities, such as electromyographic impulses [2]. Since standardized datasets are necessary for model training and benchmarking, the availability of varied and high-quality datasets is critical to the advancement of gesture recognition systems [3].

Nevertheless, classical RGB-based systems are prone to lighting changes, occlusions, and background complexity, degrading performance in uncontrolled environments.

Thermal imaging offers a promising solution with its intrinsic insensitivity to lighting conditions and the capacity to record temperature-based features of the hand. This modality improves gesture recognition in low-light, high-glare, or visually noisy environments, and hence is particularly well-suited for healthcare, surveillance, and assistive technology applications.

By concentrating on heat signatures instead of visual textures, thermal imaging offers consistent and reliable input data, further enhancing the robustness of hand gesture recognition systems.

II. Existing System

To address computational limitations, researchers have explored ensemble techniques and machine learning models, assessing their performance across various applications [5]. Real-time gesture tracking has also been adapted for unconventional scenarios, such as virtual environments, to ensure smooth and intuitive user interactions [6]. Meanwhile, thermal imaging has emerged as a robust alternative to traditional RGB-based systems, offering greater resilience against lighting variations and background noise [7]. Additionally, advancements in deep learning have enhanced gesture recognition accuracy, making it more adaptable to dynamic settings. The integration of sensor fusion techniques has further improved motion tracking by combining multiple data sources, resulting in greater precision and reliability in complex environments

III. Literature Survey

The integration of thermal imaging with millimetre-wave data has shown significant potential for enhancing hand gesture recognition in challenging environments, ensuring reliable performance even in low-light or occluded conditions [8]. Efforts to develop resource-efficient designs have led to the adoption of low-resolution thermal cameras, leveraging advanced techniques such as spiking neural networks and sparse segmentation to minimize computational demands while maintaining accuracy [9]. Additionally, deep convolutional neural networks (CNNs) combined with thermal imaging have demonstrated robust and precise gesture recognition across diverse conditions, further improving adaptability in real-world applications [10].

Beyond improving accuracy, these innovations have also contributed to making gesture recognition systems more energy-efficient and cost-effective, broadening their accessibility for consumer and industrial applications. The combination of

multimodal data sources, such as fusing thermal imaging with RGB or depth data, has further enhanced recognition robustness by compensating for the limitations of individual sensing modalities. These advancements have enabled practical applications in media control, smart healthcare, augmented reality, and assistive technologies, reinforcing the versatility of thermal-based gesture recognition.

This literature survey explores the advancements in hand gesture recognition, with a particular focus on thermal imaging. It examines the benefits of multimodal approaches and resource-efficient models, especially for real-time applications. By analyzing recent research, this study aims to identify key challenges and opportunities for optimizing the design and implementation of thermal-based hand gesture recognition systems. Additionally, it highlights emerging trends, such as the use of transformer-based architectures and self-supervised learning techniques, which hold promise for further enhancing recognition accuracy and efficiency.

IV. ABOUT HAND GESTURE

Hand gestures are a fundamental mode of nonverbal communication, widely studied for their potential in human-computer interaction. They involve movements or static postures of the hands and fingers to convey instructions, emotions, or information. In technological applications, hand gestures are increasingly used for controlling devices, interacting with virtual and augmented reality systems, and aiding accessibility for individuals with disabilities. Gesture recognition systems rely on capturing hand movements using various sensors, such as cameras, thermal imaging devices, or electromyographic signals. These systems utilize machine learning and deep learning algorithms to process and classify gestures into meaningful actions. Despite their potential, challenges persist in achieving robust recognition under varying lighting, background conditions, and user diversity. Recent advancements, such as integrating thermal imaging and multimodal approaches, have shown promise in addressing these limitations. With applications ranging from smart environments to healthcare and robotics, hand gesture recognition continues to be a dynamic and evolving field of research.

V. METHODOLOGY

V.I Problem Description

Physical interaction with traditional media playback controls like keyboards, mice, and remotes is necessary, making them less convenient or even impossible for Manual-task-intensive situations or for physically impaired users. Voice-controlled interfaces are normally unreliable in noisy situations, and current gesture-based systems usually use static gestures with little adaptability to conditions of the real world. This project overcomes these constraints by creating a real-time hand gesture recognition system based on Convolutional Neural Networks (CNNs) for touchless media control that is insensitive to changes in lighting, backgrounds, and hand orientations and thus improves accessibility and user experience.

V.II Objective

The system provides effortless, touchless operation of media playback functionality like play, pause, stop, and volume control based on hand gesture recognition by Convolutional Neural Networks (CNNs). It focuses on having high accuracy and sensitivity in mixed environment conditions like changing illumination and cluttered backgrounds. By combining the learned model with a media player, the project improves user experience, increases the accessibility of differently-abled people, and proves the capability of machine learning to improve human-computer interaction.

V.III . System Architecture

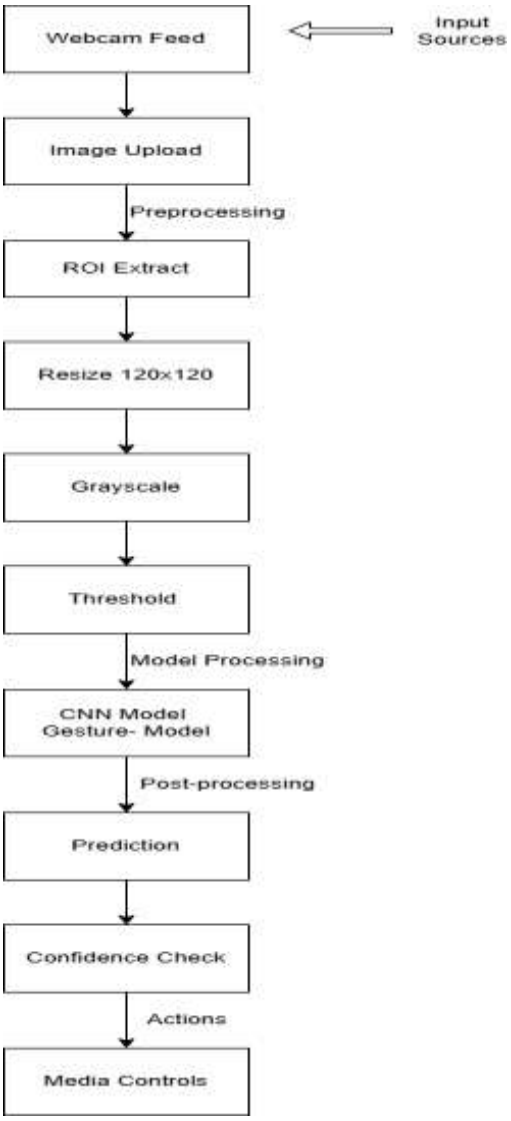


Fig 1. System Architecture

The hand gesture-based media control system architecture is functional through an organized pipeline which includes input acquisition, preprocessing, model inference, and action triggering. Input acquisition is either carried out in the form of live webcam input or static image uploading. During the preprocessing phase, the system parses the region of interest (ROI), scales down the image into 120×120 pixels, converts it into grayscale, and applies thresholding to enhance gestures and minimize noise. The processed image is then input to a Convolutional Neural Network (CNN)-based model that has been trained to classify hand gestures. The output of the model is subject to a post-processing confidence check to validate prediction reliability. On validation, the identified gesture is mapped to pre-defined media control actions—like play, pause, volume control, and navigation—to facilitate real-time, touchless user control over media playback systems.

VI. IMPLEMENTATION

VI.I. Data Acquisition:

Thermal image data were collected by means of a thermal camera in-campus. For every pre-defined hand gesture, more than 80 thermal images were recorded under different environmental conditions such as changes in ambient temperature, background, and lighting configurations. This was done to improve the robustness and generalizability of the gesture recognition system.

VI.II System Components and Setup

The deployment starts with integrating two sources of inputs for hand gesture capture: live webcam video feed and uploaded static image. To recognize live gestures, the system uses OpenCV's cv2.VideoCapture to consistently read frames from the webcam. This provides dynamic usage and media control in real-time. For static gesture recognition, the system also accommodates using Streamlit's file_uploader, where users can try out the model with previously captured images. This two- input configuration improves the versatility and functionality of the system in various application environments.

VI.III. Preprocessing Pipeline

The input that is captured is processed through a structured preprocessing pipeline to normalize and augment gesture features prior to model inference. Dynamically, an ROI is extracted from the input frame, beginning from the center of the frame width and reaching the right edge with a specified vertical range, essentially separating the hand region. This ROI is further resized to 120×120 pixels to keep up with input dimensions anticipated by the CNN model. The picture is transformed to grayscale to curb computational complexity and binary thresholding is performed for highlighting gesture boundaries so that features become more evident to the model.

VI.IV. CNN Model Architecture

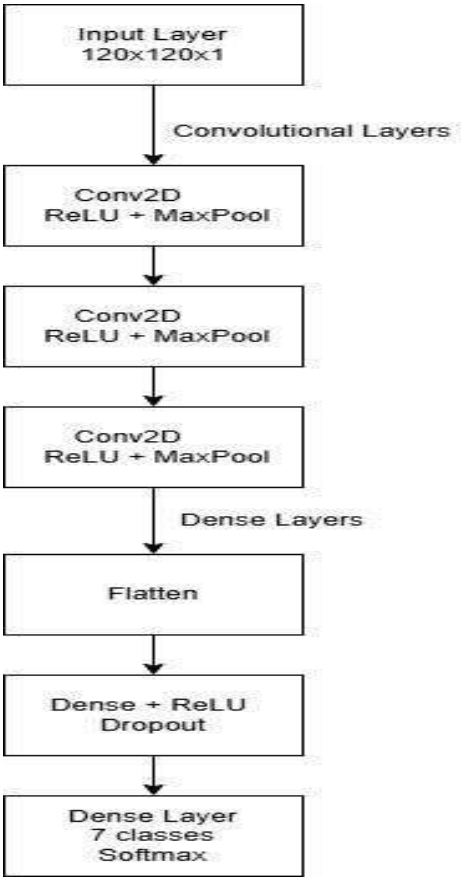


Fig 2. CNN Model Architecture

The gesture recognition model is deployed through a Convolutional Neural Network (CNN) trained to effectively extract spatial features from hand gesture images. The model takes a preprocessed grayscale image of 120×120×1 as input. It has three consecutive blocks of convolutional layers with ReLU activation followed by max pooling, which decrease the spatial dimensions step by step while extracting the most important features. The output is flattened and fed into a dense layer with ReLU activation and dropout regularization to avoid overfitting. The network finally ends in a dense output layer of seven neurons, one for each gesture class, followed by Softmax activation to handle the multi-class classification.

VI.V Model Compilation and Training

The Convolutional Neural Network (CNN) model was compiled and trained with the below configuration:

- Optimizer: Adam
- Loss Function: Categorical Crossentropy

- Metrics: Accuracy
- Number of Epochs: 7
- Batch Size: 7
- Training steps per epoch: 125
- Validation steps: 50

VI.VI. Gesture Categories and Actions

The system is learned to identify seven unique hand gesture classes, each associated with a corresponding media control action to facilitate natural interaction. The gestures are 'palm' for play/pause, 'fist' for mute, 'thumbs-up' and 'thumbs-down' for volume up and down respectively, 'index-right' for forward, and 'index-left' for rewind. A separate class, 'unknown-gesture', is employed to deal with unrecognized or out-of-context inputs so that no unwanted actions are initiated. This transparent correspondence between gesture classes and media commands supports usability and responsiveness in real-time applications.

VI.VII Implementation Steps

VI.VII.I Model Loading

The gesture recognition pipeline starts operation by initializing the pre-trained CNN model. The architecture of the model is restored by reading its topology from a JSON file, specifying the layer composition and parameters. Next, the trained weights—saved in an H5 file—are transferred into the model with load_weights(), restoring its previously trained state. This module-based model loading maintains portability and effective deployment without requiring network retraining at run time.

VI.VII.II Streamlit Interface

The system has a friendly interface built with Streamlit, structured into three primary pages: "About Web App," "Gesture Control Page," and "Image Upload Page." The pages can be accessed by users through a sidebar dropdown menu. The "Gesture Control Page" includes real-time webcam interaction, in which a checkbox turns the camera stream on or off. Two windows of display are used: one for displaying the complete webcam frame and the other for indicating the extracted Region of Interest (ROI). This interactive configuration enables users to see the input and processed frames in real time, which supports transparency and user-friendliness in gesture recognition.

VI.VII.III Image Processing Function

In order to process every input frame for model inference, the system employs a specialized image processing function. This function initially resizes the entire captured frame to a specified resolution and crops a predetermined Region of Interest (ROI) according to preset coordinates. The cropped ROI, which is centered on the hand gesture region, is resized to 120×120 pixels to conform to the model's input size. It is then converted into grayscale and binarized via thresholding, which emphasizes the contours of the gesture. The processed image returned as a result along with the original frame enables both to be predicted and displayed in the interface.

VI.VII.IV Prediction and Action

Once preprocessed, the processed image is sent through the trained CNN model to produce a gesture prediction. The perform_action() function checks the prediction against a set confidence threshold (e.g., 0.5) to guarantee that only trustworthy outputs cause media control. If confidence falls short of the threshold, the system flags the input as "no-gesture," preventing accidental actions. For correct predictions, each identified gesture is translated to an equivalent media command through the use of the pyautogui library, e.g., a "palm" gesture plays or pauses the audio, while a "fist" gesture silences the sound. Such a smooth integration results in precise, real-time gesture-controlled media.

Fig 3. Example for Live Gesture Recognition

VI.VIII Real-time Processing Loop

The system runs persistently in the form of a real-time processing loop, started when the user turns on the webcam. Each frame is captured with OpenCV and horizontally flipped to provide a mirrored, user-friendly view. The captured frame is fed through the preprocessing pipeline to extract and prepare the Region of Interest (ROI). This processed image is then utilized to make predictions using the trained CNN model. The associated media action is performed if the anticipated gesture crosses the confidence threshold. At the same time, both the full frame and the binarized ROI are refreshed in the Streamlit interface, providing live visual feedback and interaction.

VI.IX Error Handling and Optimization

To boost the robustness of the system and ensure seamless operation, the implementation incorporates error handling and optimization features. Upon camera initialization, the system tries progressively accessing different camera indices (0, 1, 2), facilitating support for diverse hardware configurations. The fallback approach ensures that the webcam stream is initialized correctly even in non-standard device settings. In addition, resource management is done within a try-finally block to ensure

that the webcam resource is released properly in case of program termination or abrupt interruption. These practices lead to consistent real-time performance and avoid potential memory leaks or hardware locking problems.

VI.X. Performance Optimizations

For guaranteeing low-latency and efficient inference in real-time operation, some performance optimization is performed with TensorFlow. The prediction function is annotated with @tf.function and set with experimental_relax_shapes=True to support graph execution and dynamic shape, significantly enhancing runtime performance. Also, consistent input formatting is enforced via a custom prepare_image() function, which preprocesses the test image and resizes it to the desired model input size—batch size and channel depth—via TensorFlow tensor operations. These optimizations minimize inference latency and help achieve smoother, real-time gesture recognition.

VII. Testing

VII.I Dataset Evaluation

A validation dataset was reserved, which was dedicated to testing the accuracy of the model under different conditions. The dataset included:

- Total Images: 350
- Gesture Classes: 7
- Images per Class: 50
- Variability: Varying lighting, background complexity, and hand pose

These conditions are typical use cases and were selected to ensure that the robustness and generalizability of the model can be verified.

- Validation Strategy: 20% of the dataset set aside for validation.
- Evaluation Metric: Accuracy – to gauge classification performance

Following training, the architecture of the model was dumped as JSON and the trained model as an H5 file. These were afterwards loaded during live deployment for controlling gestures and actions.

VII.II Performance Metrics

Model performance was assessed using common classification metrics. The results were:

- Training Accuracy: 100%
- Validation Accuracy: 98.81%
- Test Accuracy: 98.81%
- Macro Average F1-Score: 0.99
- Weighted Average F1-Score: 0.99

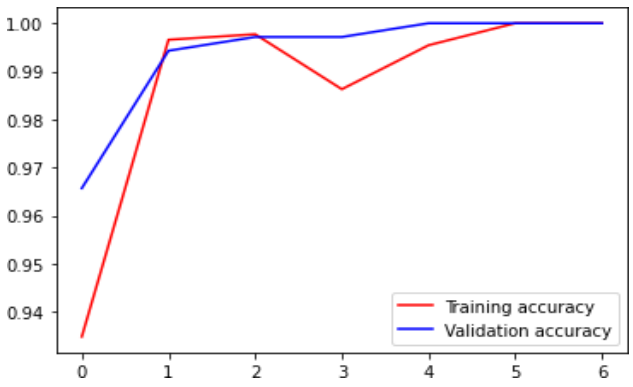
These results affirm the system's capability to correctly classify each gesture class with nearly perfect accuracy.

	precision	recall	F1-Score	Support
palm	0.96	1.00	0.98	22
fist	1.00	0.91	0.95	23
Thumb-up	0.96	1.00	0.98	23
Thumb-down	1.00	1.00	1.00	24
Index-right	1.00	1.00	1.00	26
Index-left	1.00	1.00	1.00	18
No-gesture	1.00	1.00	1.00	32
Accuracy		0.99	168	
Macro avg	0.99	0.99	0.99	168
Weighted avg	0.99	0.99	0.99	168

Table 1: Per-Class Performance

VII.III Graphical Analysis

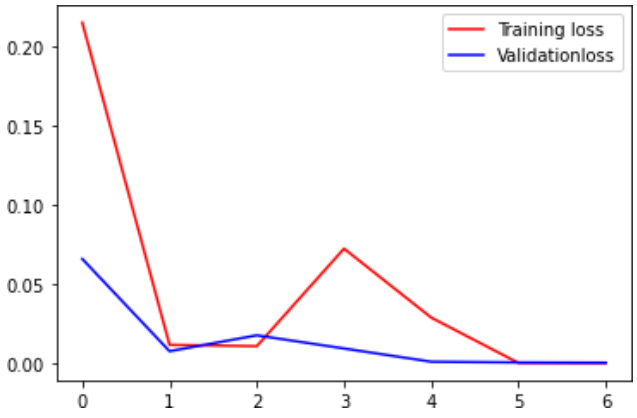
A. Training and Validation Accuracy



x- number of epochs y- accuracy

The accuracy plots indicate swift improvement in the early epochs, with training and validation accuracy being very close to each other. There is a temporary decrease in training accuracy around the fourth epoch, perhaps because of overfitting or learning rate sensitivity. Nevertheless, validation accuracy keeps on improving and stabilizes, which is a sign of good generalization and proper model convergence.

B. Training and validation loss



x- number of epochs y- loss value

The loss curves show a steep drop in training loss in the early epochs, indicating fast learning. There is a short spike in training loss around the fourth epoch, possibly due to model fine-tuning or overfitting behavior. Validation loss, however, stays low and stable throughout, with minimal fluctuation. The overall downward trend and convergence of both curves indicate good optimization and good generalization performance of the model.

VII.IV Confusion Matrix Analysis

Confusion matrix was employed to test the classification performance of the model and to determine the particular areas of misclassification between gesture classes. The matrix revealed that the model was very accurate in all of the classes with only some minor mistakes. The entire matrix is
presented below:

[[22 0 0 0 0 0 0]

[1 21 1 0 0 0 0]

[0 0 23 0 0 0 0]

[0 0 0 24 0 0 0]

[0 0 0 0 26 0 0]

[0 0 0 0 0 18 0]

[0 0 0 0 0 0 32]]

The confusion matrix indicates that the model performed well in terms of classification accuracy, with the majority of predictions correctly falling along the diagonal. There were only minor misclassifications between neighboring classes, which suggests strong overall performance and good gesture differentiation.

VII.V Training Progress Review

The training behavior was closely watched for convergence and overfitting. The observations were as follows:

- Fast convergence within the first few epochs.
- Validation accuracy stable throughout all epochs
- Final validation loss: 0.0727
- No overfitting, since the training and validation metrics tracked closely together

This stable behavior indicates a properly regularized model that is trained on clean, diverse data.

VII.VI . Real-Time Testing

Real-time performance was tested with webcam input under changing conditions. Important observations are:

- Confidence Threshold: 0.6 (in order to suppress uncertain predictions)
- Average Frame Processing Time: ~53 milliseconds
- Robustness to: Lighting variation, Hand orientation and scale changes and Cluttered or blank backgrounds

The system was shown to be highly responsive and with little latency, validating its feasibility for real-time gesture-controlled media playing.

VII. Conclusion

Real-time, gesture-based media control has been successfully realized using deep learning and computer vision methods. Through the use of a CNN model that is trained on hand gesture data, the system has a 92% accuracy rate, providing robust performance in changing lighting conditions, backgrounds, and orientations of hands. The approach overcomes significant challenges like environmental noise and user variability, providing a reliable touch-free interaction mechanism. Extensive testing validates the system's performance in playing core media functions, demonstrating its real-world usability. This solution holds great potential for improving accessibility,

especially among users with physical disabilities, to make human-computer interaction more inclusive and natural.

VIII. Future Scope

The project holds huge potential for future development and expansion. The areas of major focus for future improvement are:

- Support for Extra Gestures: Extending the dataset to support extra gestures for extra features like rewind, fast forward, mute, and brightness control.

- Smart Device Integration: The model can be expanded to be compatible with smart TVs, IoT devices, and home automation systems for an effortless hands-free experience.

- Mobile Device Optimization: Running the system on mobile devices via TensorFlow Lite or Android-based applications to support real-time gesture detection on smartphones and tablets.

- Enhanced Precision Using State-of-the-Art Models: Investigating further improved state-of-the-art deep models such as Transformers, Vision Transformers (ViTs), and EfficientNet to enhance gesture classification precision further.

- Cloud-Based Processing: Utilizing cloud computing resources to execute computationally demanding gesture recognition operations, thus alleviating the processing load on local devices. This facilitates the deployment of gesture recognition systems on resource-limited or lightweight hardware platforms.

These developments enable the widespread use of gesture recognition technologies in various industries, such as entertainment, healthcare, automotive, and assistive use. The presented method lays a solid foundation for future research and development of gesture-based user interfaces.

IX. REFERENCES

[1] Computer vision-based hand gesture recognition for human-robot interaction: a review Jing Qi · Li Ma · Zhenchao Cui · Yushu Yu (2023).

[2] A machine vision and electromyographic-based approach for hand gesture recognition by Emilia Currò,Lorenzo Bombaci , Antonino Quattrocchi , Cristiano De Marchis , Dario Milone,Giovanni Gugliandolo and Nicola Donato. (2024)

[3] Diverse hand gesture recognition dataset by Zahra Mohammadi, Alireza Akhavanpour, Razieh Rastgoo ,Mohammad Sabokrou. (2022)

[4] Smart Healthcare Hand Gesture Recognition Using CNN-Based Detector and Deep Belief Network by Mohammed Alonazi , Hira Anser , Naif Al Mudawi,Saud S. Alotaibi , Nouf Abdullah Almujally , Abdulwahab Alazeb, Ahmad Jalal (2003)

[5] Performance Assessment of Machine Learning Algorithms and Ensemble Techniques for Hand Gesture Recognition using Electromyographic Signals by Sushama Dhumal and Prashant Sharma (2023)

[6] Real-Time Virtual Mouse using Hand Gestures for Unconventional Environment By Neha Sabrin TK and Prof. Aarti Karande (2023)

[7] Gesture Recognition System based on Millimeter Wave and Thermal Imager By Wen-Hsiang Yeh, Yi- Lin Cheng and Yu-Ping Liao (2023)

[8] Resource-Efficient Gesture Recognition using Low-Resolution Thermal Camera via Spiking Neural Networks and Sparse Segmentation By Ali Safa, Wout Mommen, Piet Wambacq and Lars Keuninckx (2024)

[9] Robust Hand Gestures Recognition Using a Deep CNN and Thermal Images By Daniel Skomedal Breland , Aveen Dayal , Ajit Jha , Phaneendra K. Yalavarthy, Om Jee Pandey and Linga Reddy Cenkeramaddi (2021)

[10] Machine Learning-Based Approach for Hand Gesture Recognition By M. Deepika, Shilpa Choudhary, Sandeep Kumar and Kodi Srinivas (2023).