HAND GESTURE RECOGNITION

A report on Computer Vision Lab Project
[CSE-3181]

Submitted By
<Veera Venkata Satvik Yerra & 210962042 >
<Dantuluri Venkata Neerajh Varma & 210962076>
<Abhiram Varma & 210962072>

MANIPAL
ACADEMY of HIGHER EDUCATION
(Institution of Eminence Deemed to be University)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MANIPAL INSTITUTE OF TECHNOLOGY,
MANIPAL ACADEMY OF HIGHER EDUCATION

< NOVEMBER 2025>

HAND GESTURE RECOGNITION

Satvik Yerra, Neerajh Varma, Abhiram varma

CSE (AI and ML), Manipal Institute of Technology, India

satvikyvv@gmail.com, sheldorabhiram@gmail.com, neerajhvarma999@gmail.com

***Abstract*—**

Hand gesture recognition is a valuable technology with a wide range of applications, from sign language interpretation to human-computer interaction. This abstract focuses on implementing hand gesture recognition using OpenCV and Python. OpenCV, a powerful computer vision library, provides the tools to capture, preprocess, and analyze video frames. By leveraging its capabilities, we can develop a system that detects and recognizes hand gestures in real-time.. We discuss the challenges and potential applications of hand gesture recognition in this project. The combination of OpenCV and Python offers a versatile platform for building gesture recognition systems, making it a promising area of research and development.

***Keywords*—**

Computer vision, Gesture Recognition, python, training data, data collection

INTRODUCTION

In the realm of computer vision and human-computer interaction, hand gesture recognition has emerged as a fascinating and practical technology. This project utilizes OpenCV, Python, and libraries such as TensorFlow and Keras to create a robust hand gesture recognition system. By training a machine learning model with carefully curated data, the system can detect and interpret hand gestures in real-time via a live camera feed. It not only recognizes these gestures but also provides immediate feedback by displaying their names. This integration of cutting-edge technologies has a wide range of applications, from interactive gaming to sign language interpretation. In this project, we explore the techniques that make this possible, demonstrating the fusion of software and hardware for an immersive and interactive experience.

LITERATURE REVIEW

Some of the well-known research papers on hand gesture recognition are:

1. “Real-time hand gesture recognition using Haar-like features” – Navneet Dalal and Bill Triggs:

   The paper focuses on real-time hand gesture recognition using Haar-like features and a webcam. The authors propose a system that can detect and recognize gestures, making it suitable for interactive applications and human-computer interaction. Haar-like features are used for feature extraction and classification.

   The key features of this project are: Real-time processing, simple feature extraction, and efficiency for real-time applications.

2. “Hand gesture recognition based on a novel shape descriptor” - Q. Huang, S. Ji, Y. Xu, C. Liu:

This paper presents a novel shape descriptor for hand gesture recognition. The descriptor is designed to capture the shape information of hand gestures, making it suitable for a wide range of gestures

The key features of this project are: Robust to changes in lighting and viewpoint, capable of recognizing various hand gestures based on shape features.

3. "Real-time American Sign Language recognition using a webcam - N. Kapur, D. Tao, T. Huang:

   This research paper focuses on real-time American Sign Language (ASL) recognition. It proposes a system that can recognize ASL signs using OpenCV and a standard webcam, making it a valuable tool for the hearing-impaired community.

   The key features of this project are: Real-time ASL recognition, low-cost solution using readily available hardware.

4. "Hand gesture recognition for human-computer interaction"-Vladislav Bidikov, Vladimir Trajkovik:

   This paper discusses the application of hand gesture recognition for human-computer interaction. It explores various techniques and algorithms used in OpenCV for recognizing hand gestures and their applications in interactive systems.

   The key features of this project are: Provides a comprehensive overview of hand gesture recognition for human-computer interaction, highlighting its potential in various applications.

5. "A survey of vision-based hand gesture recognition"- Li Yang, Jun Guo, Shengyong Chen, Mingui Sun:

   This paper offers a comprehensive overview of vision-based hand gesture recognition techniques, covering various methods, feature extraction, recognition algorithms, and applications. It also addresses challenges and suggests future research directions.

   The key features of this project are: Offers detailed insights into different methodologies and identifies challenges and future research areas.

METHODOLOGY

As we know the main aim of this project is to identify hand gestures, some examples of hand gestures the project aims to detect are:

Some hand signs such as : 1 2 3 4 5

The project relies on the OpenCV library for computer vision tasks, and the **cvzone** and **cvzone.ClassificationModule** libraries for hand detection and hand gesture classification. The code implemented in the project is divided into two parts: one for capturing images of hand gestures and another for recognizing and classifying those gestures.

Part 1 : capturing and saving hand gesture images

- Camera setup - The code initializes the computer's camera using OpenCV (**cv2.VideoCapture(0)**)
- Hand Detection - The **HandDetector** class from the **cvzone** library is used to detect and track the user's hand in the camera feed.
- Image processing - The detected hand is cropped and resized to a fixed size (**imgSize x imgSize**), where **imgSize** is set to 300.

  The project accounts for variations in hand aspect ratios, ensuring that the resized image maintains the correct proportions.
- User Interaction – The camera feed and the processed images are continuously displayed and visible to the user

  When the 's' key is pressed, the current processed hand image is saved to a specified folder (with the images that will be used for model training) with a unique filename that includes a timestamp.

  A counter keeps track of the number of saved images.

This part includes everything about the capturing of the hand images that will be used for training in the next part

The project uses a pre-trained Keras model for hand gesture classification and does not include the model training process. The relevant parts regarding TensorFlow, Keras, and the pre-trained model used in the project are:

TENSORFLOW -- TensorFlow is an open-source machine learning framework developed by Google. It is widely used for various machine learning and deep learning tasks, including building and training neural networks. TensorFlow is used as a backend for Keras, a high-level neural networks API that runs on top of TensorFlow.

KERAS -- Keras is a popular high-level neural networks API that provides a user-friendly interface to build, train, and deploy neural networks. It can run on top of various deep learning frameworks, including TensorFlow. In this project, Keras is used for loading the pre-trained model and making predictions on the hand gesture images.

From the captured images initially we have multiple datasets namely 0, 1, 2 ,3, 4, 5, call, please, rockandroll, shoot, thumbs_down, thumbs_up.

We use teachable machine from google to train the model with the previously captured images, it is a TensorFlow model and then after this we convert this model into a keras .h5 model and now this trained model is imported back into the main code and used for gesture identification

Part 2: Recognizing the hand gestures

- Importing libraries and models: This part imports the necessary libraries, including OpenCV, cvzone, and cvzone.ClassificationModule. Additionally, it loads a pre-trained hand gesture classification model (**keras_model.h5**) and the associated gesture labels (**labels.txt**).

- Camera setup: Similar to Part 1, the code initializes the computer's camera to capture real-time hand gestures.

- Hand Detection: Once again, the **HandDetector** class is used to detect and track the user's hand in the camera feed

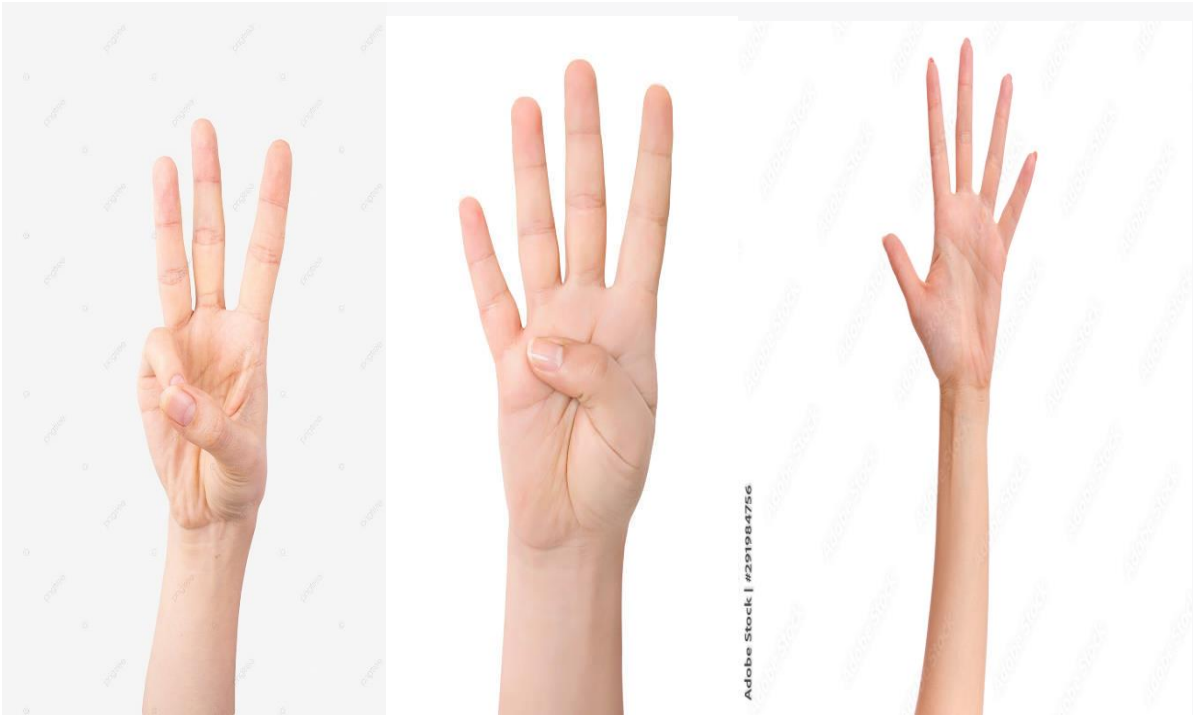
- Image cropping and resizing: The detected hand is cropped and resized to match the same dimensions (300x300 pixels) as in the first part. This ensures that the images used for recognition are consistent with the training images.

- Gesture classification: A pre-trained model, loaded from "Model/keras_model.h5," is used for gesture classification. This model has been trained separately to recognize hand gestures. The classification model predicts the gesture based on the resized hand image, and the predicted label is obtained. The code overlays rectangles and text on the video frame to highlight the detected hand and display the recognized gesture. The recognized gesture label is displayed above the hand. Additionally, a colored rectangle with text indicates the recognized gesture's label on the video frame

All the gestures that the project recognizes are as follows:

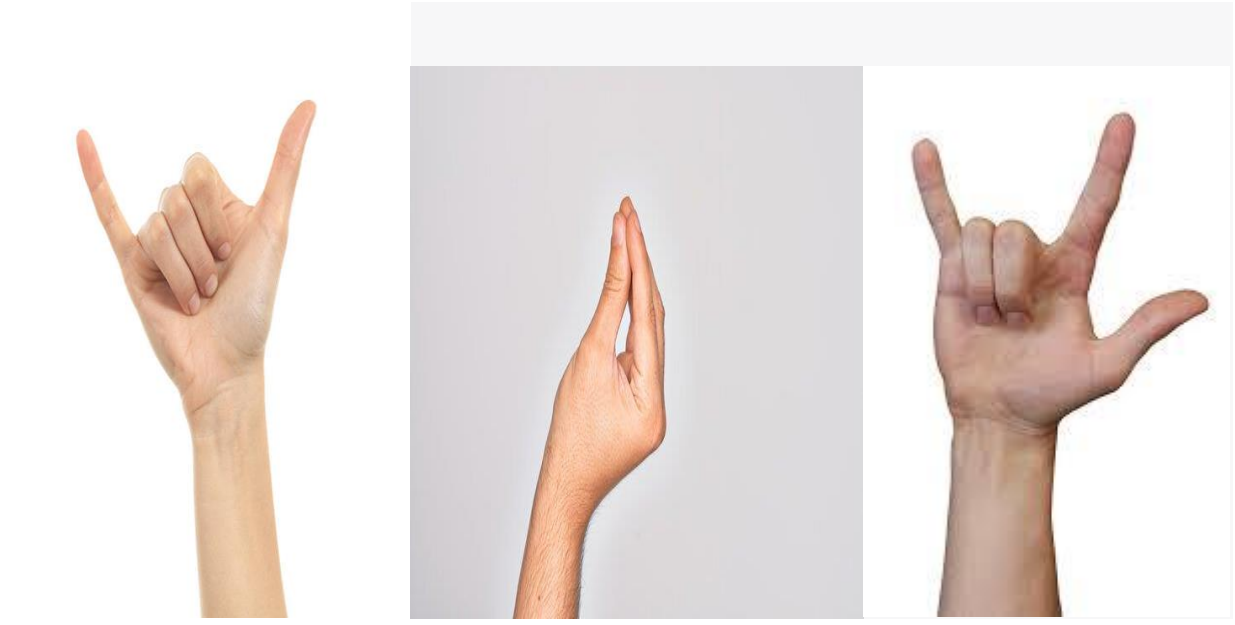

The above three images correspond to :

- 0
- 1
- 2



The above three images correspond to :

- 3
- 4
- 5



The above three images correspond to:

- Call
- Please

- Rock and roll

The above three images correspond to:

- Shoot
- Thumbs down
- Thumbs up

Overall, the methodology involves real-time hand gesture detection, preprocessing of hand images, and their classification using a pre-trained model. The recognized gestures are displayed on the video feed in real time.

EXPERIMENTAL SETUP

The experimental setup involves a computer with a webcam, the OpenCV library for computer vision tasks, and two Python scripts utilizing the **cvzone** and **Classifier** modules for hand tracking and gesture recognition.

HARDWARE –
A computer with a webcam is used to capture live video frames.

SOFTWARE-
OpenCV, a computer vision library, is used for real-time video frame capture and image processing.

The **cvzone** library provides functions for hand tracking and gesture recognition.

A custom-trained classifier is utilized for gesture recognition based on a pre-trained model and labels.

The data collection procedure involves capturing live video frames from the computer's webcam and processing them to detect and recognize hand gestures.

The captured images are used to train the model using TensorFlow and keras and then finally the gestures will be recognized from the live camera feed

The procedure can be summarized as follows:

- The **cv2.VideoCapture** function is used to access the computer's webcam and capture video frames.
- The **HandDetector** class from the **cvzone** library is employed to track and detect hands in each frame

- For the first part, the program captures and processes hand images. It creates a white background, crops the hand region, and resizes it to a fixed size for consistency.

- For the second, in addition to hand detection, the program employs a custom-trained classifier to recognize hand gestures based on processed images of the hand.

- The recognized gestures are displayed on the screen in real-time using OpenCV.

- Data collection, in this context, is the real-time processing of video frames and the classification of hand gestures.

The experimental setup focuses on real-time hand tracking and gesture recognition using computer vision techniques. It does not involve specialized instruments or detectors but rather relies on the capabilities of the computer system and the specified software libraries.

RESULTS AND DICUSSIONS

As discussed above the project has 2 parts, the first part involves capturing the images and the second part involves identifying them with the help of trained data.

Now let us see the results of the first part:

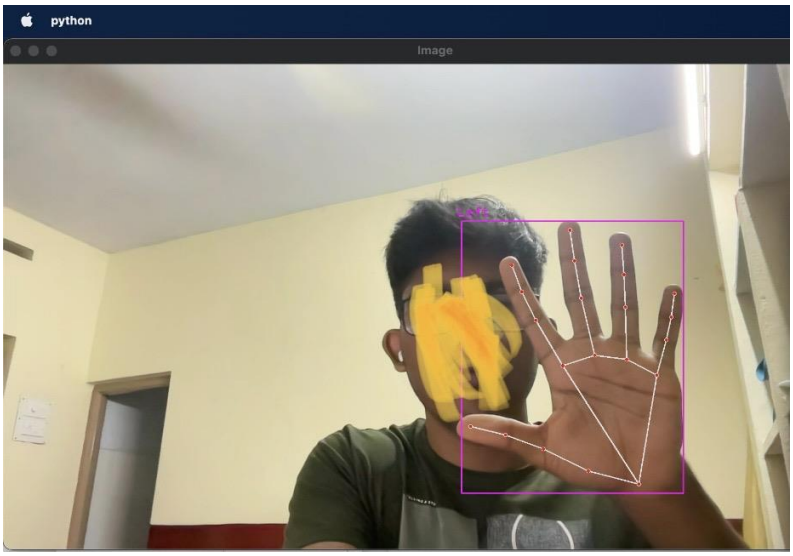
After executing the first part of the code which involves the data collection, the points on the hand are marked and the hands in the live camera feed are detected and displayed.

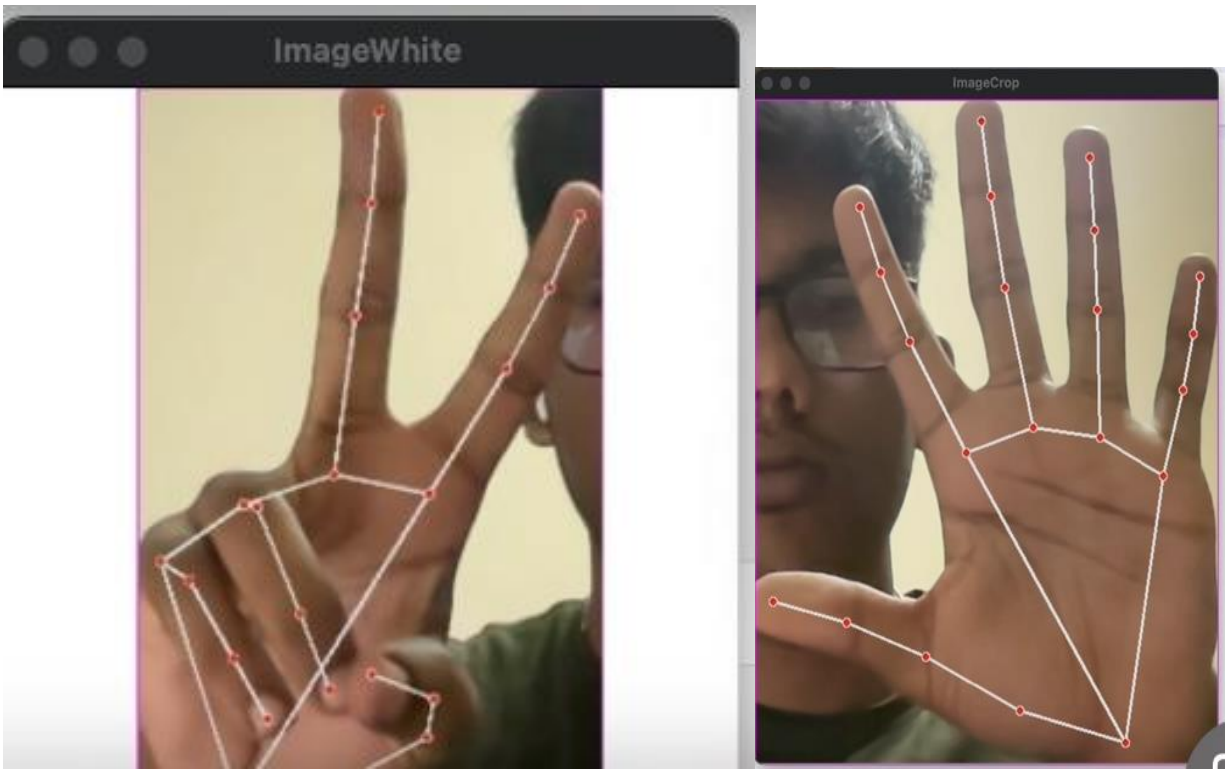
Other than the live camera feed there are 2 additional live feeds namely ‘ImageCrop’ and ‘ImageWhite’

- **imgCrop** is the cropped region of interest (ROI) of the original camera frame that contains the user's hand. This cropped region focuses specifically on the hand and is used for further processing and feature extraction.

- **imgWhite** is a blank white image (all white pixels with intensity 255) with a fixed size (300x300 pixels). It serves as a canvas or destination image for placing the cropped hand image (**imgCrop**) after preprocessing.

The output after implementation of part 1 is as follows:





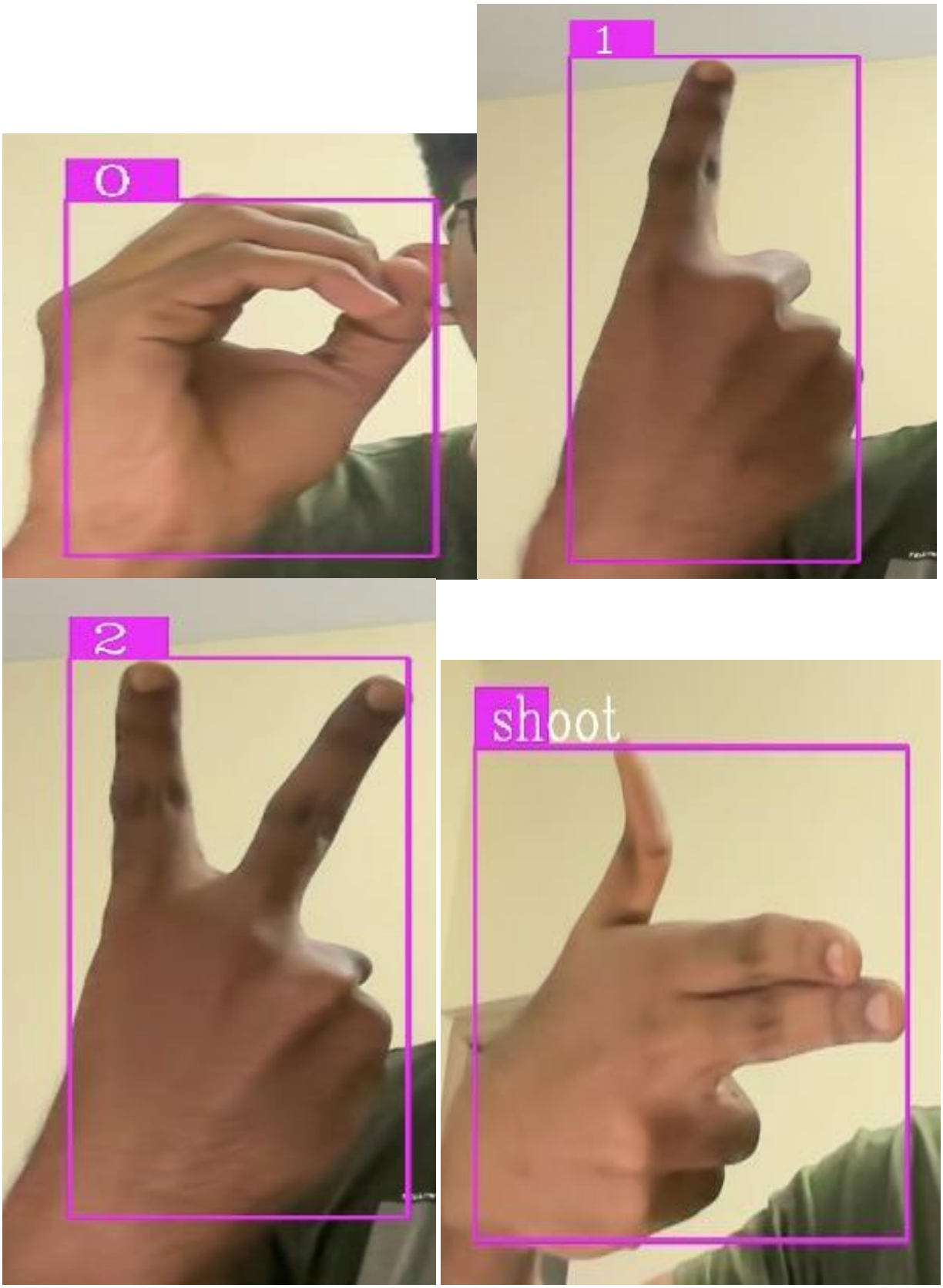
Now here upon pressing the ‘s ‘ key the images are captured and then saved into separate directories according to their respective labels.

Now
The project uses a pre-trained Keras model for hand gesture classification.

We use teachable machine from google to train the model with the previously captured images, it is a TensorFlow model and then after this we convert this model into a keras .h5 model and now this trained model is imported back into the main code and used for gesture identification

Now we implement the 2nd part of the code which involves identifying the gesture from the live camera feed with the help of the trained data.

Some of The final output where the hand gestures are identified are as follows:



Above 4 pictures depict the pictures of the live camera feed while detecting the hand gestures

corresponding to:

0 , 1 , 2 and shoot

CONCLUSION

This project represents a comprehensive solution for hand gesture recognition, covering the crucial stages of data collection and real-time recognition. By capturing and saving hand gesture images during the data collection phase, it lays the groundwork for training a robust recognition model. Once trained, the model demonstrates its versatility by offering real-time recognition of a wide range of gestures.

As the project's success hinges on effective model training and dataset preparation, it underscores the importance of these foundational steps in developing a reliable hand gesture recognition system. Furthermore, the code's extensibility allows for potential enhancements, such as expanding the set of recognized gestures or optimizing recognition accuracy. In essence, this project represents a promising contribution to the realm of computer vision and human-computer interaction. It not only addresses the technical aspects of hand gesture recognition but also opens up opportunities for further innovation and application in diverse real-world scenarios.

FUTURE WORK

The project on hand gesture recognition provides a solid foundation for further research and development

Some ways in which gesture recognition can be used in the future are:

Gesture Customization: Allow users to define and customize their own gestures for specific applications

Accessibility Features: Develop accessibility features for individuals with disabilities, such as gesture-based communication systems for those with limited mobility.

Privacy and Security: Address privacy and security concerns, especially if the system is deployed in applications involving sensitive data or access control.

Feedback Mechanisms: Incorporate feedback mechanisms, such as haptic feedback or auditory cues, to enhance user experience and provide confirmation of recognized gestures.

REFERENCES

- gesture recognition-principles, techniques and abbreviations by Amit konar and Sriparna Saha
- https://ieeexplore.ieee.org/Xplore/home.jsp
- https://developers.google.com/mediapipe/solutions/vision/gesture_recognizer/python
- "Introduction to Hand Gesture Recognition" by G. N. Purohit and S. K. Kinlekar in *International Journal of Scientific & Engineering Research*, 2012
- "Computer Vision: Algorithms and Applications" by Richard Szeliski
- "Dynamic Hand Gesture Recognition Using Convolutional Neural Networks" by Y. Li and Y. Kim in *IEEE Access*, 2018
- http://ir.juit.ac.in:8080/jspui/bitstream/123456789/7027/1/Hand%20Gesture%20Recognition%20using%20OpenCV.pdf

- “Real-time hand gesture recognition using Haar-like features” – Navneet Dalal and Bill Triggs

- “Hand gesture recognition based on a novel shape descriptor” - Q. Huang, S. Ji, Y. Xu, C. Liu:
- "Real-time American Sign Language recognition using a webcam - N. Kapur, D. Tao, T. Huang
- "Hand gesture recognition for human-computer interaction"-Vladislav Bidikov, Vladimir Trajkovik
- "A survey of vision-based hand gesture recognition"- Li Yang, Jun Guo, Shengyong Chen, Mingui Sun:

CONTRIBUTIONS

SATVIK:

- Designed the project architecture and system flow.
- Conducted research on hand gesture recognition techniques.
- Implemented hand detection, tracking, recognition using OpenCV and cvzone

NEERAJH:

- Collected and organized the dataset of hand gesture images and videos.
- Labeled the data with appropriate gesture categories and ensured quality and consistency
- Trained and fine-tuned the machine learning model for gesture classification.

ABHIRAM:

- Developed image preprocessing and augmentation techniques.
- Made sure the pre trained model was proper and efficient
- Identified and reported bugs and issues for resolution.

Everyone has contributed actively to the project and this report is made on the results we reached during the implementation of the project.