MAJOR PROJECT SEMINAR REPORT

"HAND GESTURE RECOGNITION"

University School of Information, Communication and Technology

Guru Gobind Singh Indraprastha University, Delhi

Submitted by: YASH ARYAN (06816403220)

Batch: B.Tech (CSE) 8th Semester

Mentor: Dr. Priyanka Bhutani

TABLE OF CONTENT

Certificate

This is to certify that Yash Aryan (Enrollment No.06816403220) has successfully submitted the Major Project report entitled "Hand Gesture Recognition" in partial fulfillment for the requirement of the award of the degree of Bachelor of Technology in Computer Science Engineering at USICT. The project has been carried out under my supervision and guidance. He has shown dedication, perseverance, and a willingness to learn new technologies throughout the project.

Dr. Priyanka Bhutani

University School of Information, Communication & Technology, USICT_GGSIPU, New Delhi-110078

Date: 13-05-2024

Candidate's Declaration

I, Yash Aryan (Enrollment No.06816403220), a student of B.Tech (CSE 8th Semester), USICT, Guru Gobind Singh Indraprastha University, hereby declare that the work which is presented in this Major Project Seminar Report entitled "Hand Gesture Recognition" is an original and authentic work of mine under the technical guidance of Dr. Priyanka Bhutani, Assistant Professor, USIC&T. I declare that the work in this project has not been submitted in full or in any part for any diploma or degree course of this or any other University to the best of my knowledge and belief. I will be solely responsible myself for any copyright infringement or plagiarism, if any, in the said work, and declare that all necessary due acknowledgement has been made in the content of said work. My supervisor/ guide shall not be held responsible for full or partial violation of copyright or intellectual property rights or any type of plagiarism involved above in the said work.

Name: Yash Aryan

Enrolment Number: 06816403220

Course: B. Tech CSE 8th Semester, University School of Information, Communication & Technology, USICT_GGSIPU, New Delhi-110078

Date: 24-03-2024

Acknowledgement

It gives me immense pleasure to take this opportunity to acknowledge my obligation to my mentor, Dr. Priyanka Bhutani, University School of Information and Communication Technology, GGSIPU, who has not only guided me throughout the project but also made a great effort in making the project a success. I am highly thankful to my guide for her keen interest, valuable guidance, technical acumen, round the clock encouragement, moral support & suggestions in the completion of the project.

Name: Yash Aryan

Enrolment Number: 06816403220

Course: B. Tech CSE 8th Semester, University School of Information, Communication & Technology, USICT_GGSIPU, New Delhi-110078

Date: 24-03-2024

Abstract

This project uses Mediapipe, OpenCV and Tensorflow for recognizing the hand gestures. This project was made using Python. When a user makes a gesture, it detects the hand gesture, recognizes it and displays the frames per second along with the detected gesture to the user.

The objectives of making a hand gesture recognition project using Mediapipe and OpenCV are:

1. Real-time Gesture Recognition: Developing a system capable of accurately recognizing hand gestures in real-time from camera feed.
2. Gesture Classification: Building a model that can classify different hand gestures into predefined categories or commands.
3. Human-Computer Interaction: Enabling natural and intuitive interactions between humans and computers or devices through hand gestures.
4. Accessibility: Creating interfaces that allow users with disabilities or limitations to interact with technology more easily through gestures, without relying solely on traditional input methods like keyboards or mice.

Introduction

Hand gesture recognition is a vital component in human-computer interaction, offering a natural and intuitive means of communication. In various domains such as sign language translation, virtual reality, robotics, and gaming, accurate recognition of hand gestures can significantly enhance user experience and accessibility. However, building an efficient hand gesture recognition system poses several challenges:

1. Complexity of Hand Gestures: Hand gestures can vary widely in terms of complexity, shape, and movement patterns, making their recognition a non-trivial task. Capturing the subtle nuances of hand movements and accurately translating them into meaningful commands require sophisticated algorithms.

2. Variability in Lighting and Background: Lighting conditions and background clutter can significantly affect the performance of hand gesture recognition systems. Variations in illumination and diverse backgrounds can obscure hand features, leading to errors in gesture classification.

3. Real-Time Processing: Many applications of hand gesture recognition, such as virtual reality gaming or human-robot interaction, demand real-time processing capabilities. Achieving low-latency recognition while maintaining high accuracy is essential for seamless user interaction.

4. Data Acquisition and Annotation: Acquiring a diverse dataset of hand gestures encompassing different hand shapes, orientations, and movements is crucial for training robust machine learning models. Additionally, annotating these datasets with accurate labels requires considerable effort and expertise.

5. Model Generalization: Ensuring that the trained gesture recognition model generalizes well to unseen data and can accurately classify gestures performed by different individuals is vital for its practical usability across various user demographics.

This project can accurately classify a wide range of hand gestures in real-time, under varying environmental conditions.

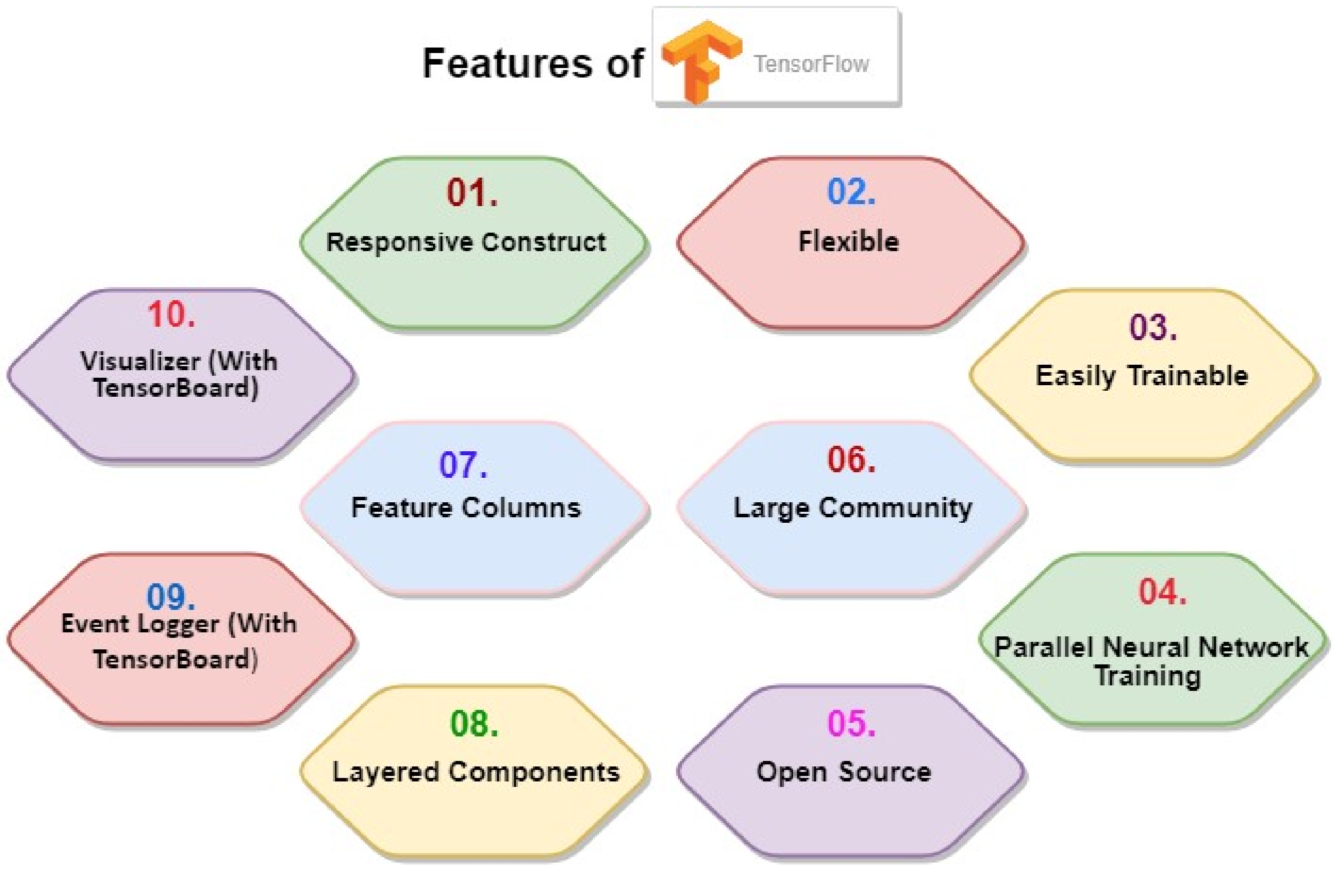
Tensorflow

TensorFlow is an open-source machine learning framework developed by Google Brain for building and training neural network models. It provides a flexible and scalable platform for a wide range of machine learning tasks, including image classification, natural language processing, and reinforcement learning. TensorFlow supports both high-level APIs for rapid development and low-level APIs for fine-grained control over model architecture and training.

Components of Tesnorflow:

- **Tensor**: A tensor is a vector or a matrix of n-dimensional that represents all type of data. All values in a tensor hold similar data type with a known shape. The shape of the data is the dimension of the matrix or an array.
- **Graphs**: TensorFlow makes use of a graph framework. The chart gathers and describes all the computations done during the training.
- **Session**: A session can execute the operation from the graph. To feed the graph with the value of a tensor, we need to open a session. Inside a session, we must run an operator to create an output.

Features of Tensorflow:



1. Responsive Construct: We can visualize each part of the graph, which is not an option while using **Numpy** or **SciKit**.
2. Flexible: It is one of the essential TensorFlow Features according to its operability. It has modularity and parts of it which we want to make standalone.
3. Easily Trainable: It is easily trainable on CPU and for GPU in distributed computing.
4. Parallel Neural Network Training: TensorFlow offers to the pipeline in the sense that we can train multiple neural networks and various **GPUs**, which makes the models very efficient on large-scale systems.
5. Large Community: Google has developed it, and there already is a large team of software engineers who work on stability improvements continuously.
6. Open Source: The best thing about the machine learning library is that it is open source so anyone can use it as much as they have internet connectivity.

Advantages of Tesnorflow:

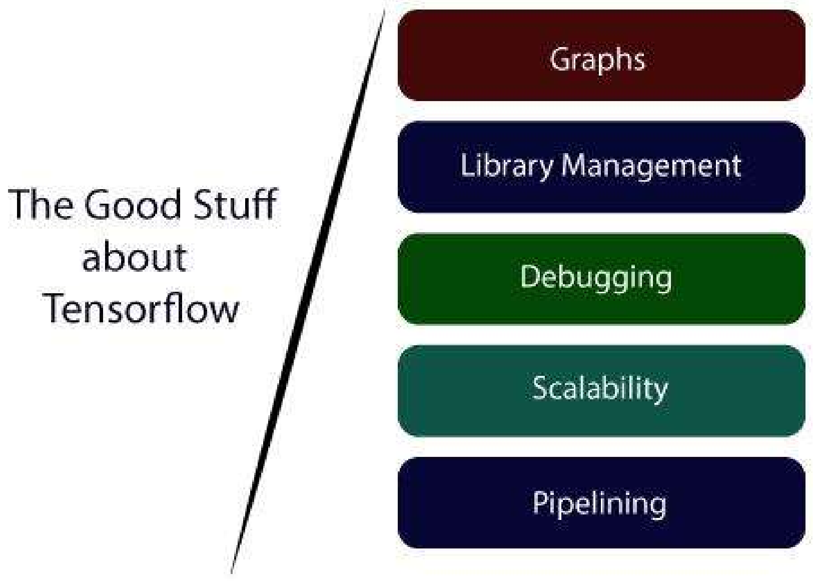
1) Graphs: TensorFlow has better computational graph visualizations.

2) Library management: It has the advantages of seamless performance, quick updates, and frequent new releases with new features.

3) Debugging: It helps us execute subpart of a graph which gives it an upper hand as we can introduce and retrieve discrete data

4) Scalability: The libraries are deployed on a hardware machine, which is a cellular device to the computer with a complex setup.

5) Pipelining: TensorFlow is designed to use various backend software (GPUs, ASIC), etc. and also highly parallel.



Disadvantages of Tensorflow:

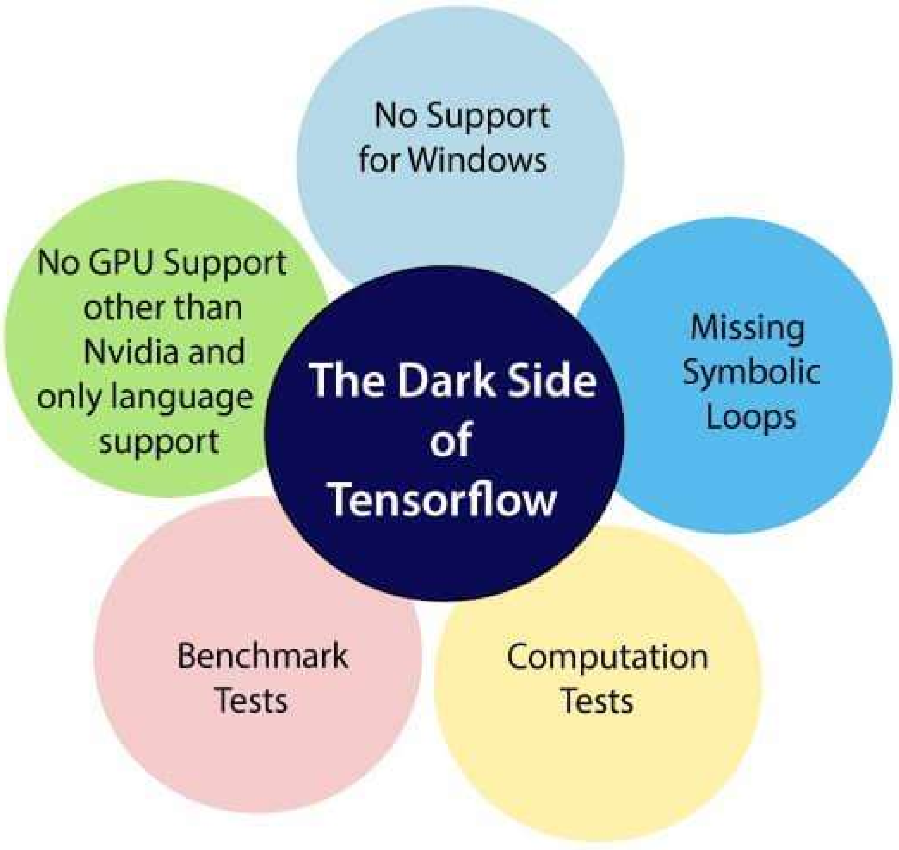
1) Missing Symbolic loops: TensorFlow does not offer functionality, but finite folding is the right solution to it.

2) No supports for windows: There is a wide variety of users who are comfortable in a window environment rather than Linux, and TensorFlow doesn't satisfy these users.

3) Benchmark tests: TensorFlow lacks in both speed and usage when it is compared to its competitors.

4) No GPU support for Nvidia and only language support: Currently, the single supported GPUs are **NVIDIA** and the only full language support of Python, which makes it a drawback as there is a hike of other languages in deep learning as well as the **Lau**.

5) Computation Speed: This is a field where TF is delaying behind.


No Support for Windows
Missing Symbolic Loops
Computation Tests
Benchmark Tests
No GPU Support other than Nvidia and only language support
The Dark Side of Tensorflow

OpenCV

OpenCV is an open-source computer vision and machine learning software library. It provides a comprehensive suite of functions and algorithms for real-time image processing, feature detection, object recognition, and more. OpenCV is written in C++ and has bindings available for Python, Java, and other languages, making it widely accessible across different platforms and development environments.

Key Features of OpenCV:

1. Image Processing: OpenCV offers a wide range of image processing functions, including filtering, transformation, edge detection, and morphological operations.
2. Object Detection and Recognition: OpenCV includes pre-trained models and algorithms for object detection, face recognition, text recognition, and other tasks.
3. Camera Calibration: OpenCV provides tools for camera calibration, stereo vision, and 3D reconstruction, enabling accurate geometric analysis of images and videos.
4. Machine Learning: OpenCV integrates with machine learning libraries such as TensorFlow and PyTorch, allowing users to build custom models for tasks like classification, regression, and clustering.
5. Real-Time Applications: OpenCV is optimized for real-time performance, making it suitable for applications such as augmented reality, robotics, surveillance, and autonomous vehicles.

Mediapipe

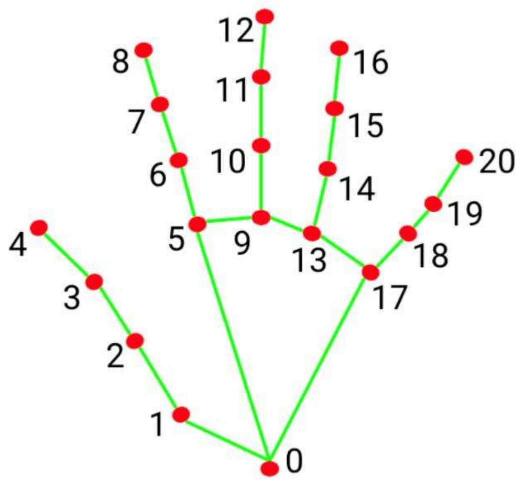
MediaPipe is an open-source framework developed by Google for building real-time multimedia processing pipelines. It provides a set of pre-built components and tools for tasks such as hand tracking, pose estimation, face detection, and gesture recognition. MediaPipe is designed to be modular and flexible, allowing developers to easily construct complex pipelines by connecting individual components.

Key Features of MediaPipe:

1. Modularity: MediaPipe pipelines are composed of modular components called calculators, each responsible for performing a specific task such as feature extraction or inference.
2. Performance: MediaPipe is optimized for real-time performance on various hardware platforms, including CPUs, GPUs, and accelerators like Google's Edge TPU.
3. Cross-Platform Support: MediaPipe supports deployment on multiple platforms, including desktop computers, mobile devices, and edge devices, enabling developers to create applications for a wide range of devices.
4. Pre-Trained Models: MediaPipe includes pre-trained models for common tasks such as hand tracking and pose estimation, allowing developers to quickly integrate advanced features into their applications.
5. Customization: MediaPipe provides tools for customizing and extending pipelines with new components, enabling developers to tailor the framework to their specific requirements.

Hand Landmark Model

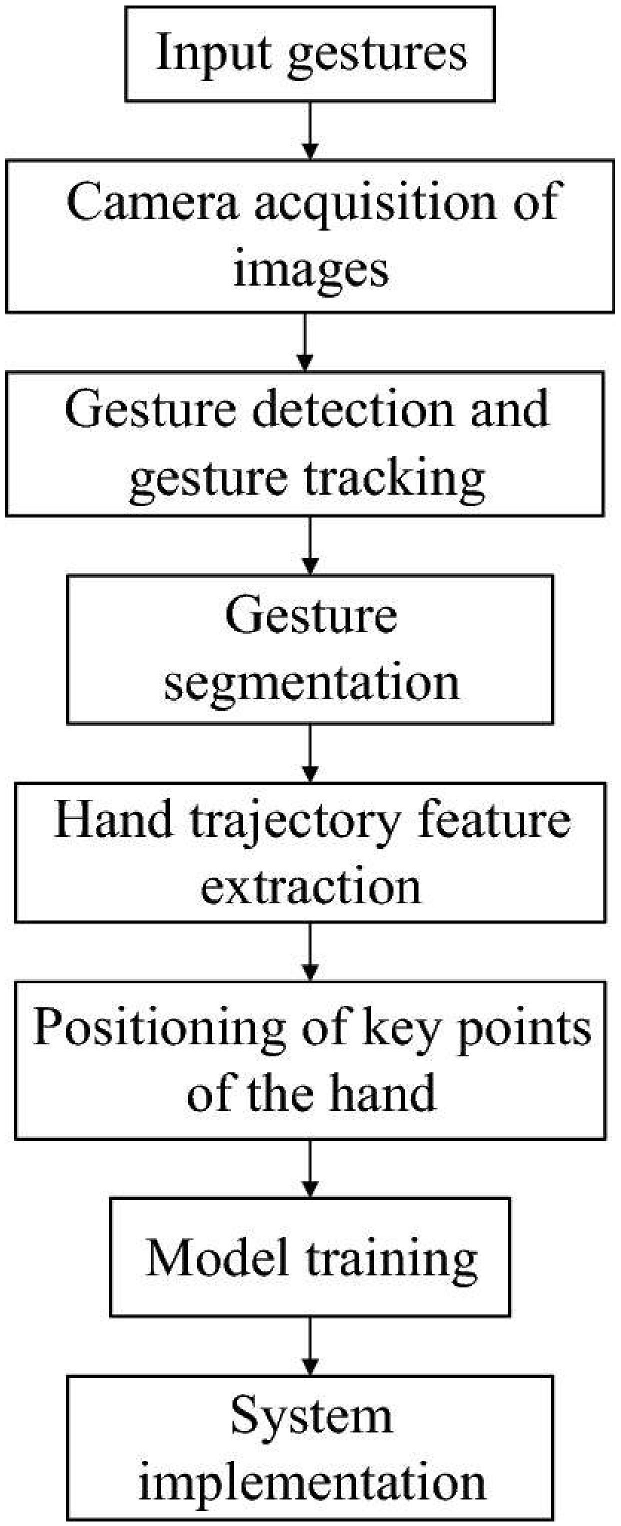
The hand landmark model bundle detects the keypoint localization of 21 hand-knuckle coordinates within the detected hand regions. The model was trained on approximately 30K real-world images, as well as several rendered synthetic hand models imposed over various backgrounds.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

The hand landmarker model bundle contains a palm detection model and a hand landmarks detection model. The Palm detection model locates hands within the input image, and the hand landmarks detection model identifies specific hand landmarks on the cropped hand image defined by the palm detection model.

Workflow


Input gestures
Camera acquisition of images
Gesture detection and gesture tracking
Gesture segmentation
Hand trajectory feature extraction
Positioning of key points of the hand
Model training
System implementation

REFERENCES

[1]Paulo Trigueiros, "Computer Vision and Machine Learning based Hand Gesture Recognition", 2015

[2]Geron Aurelien, "*Hands-On Machine Learning with Scikit-Learn*, *Keras, and TensorFlow*", 2022

[3]Nishant Shukla, "*Machine Learning with TensorFlow*", 2018

[4]Adrian Kaehler, "Learning OpenCV", 2008

[5]Andreas Muller, "*Introduction to Machine Learning with Python*", 2016