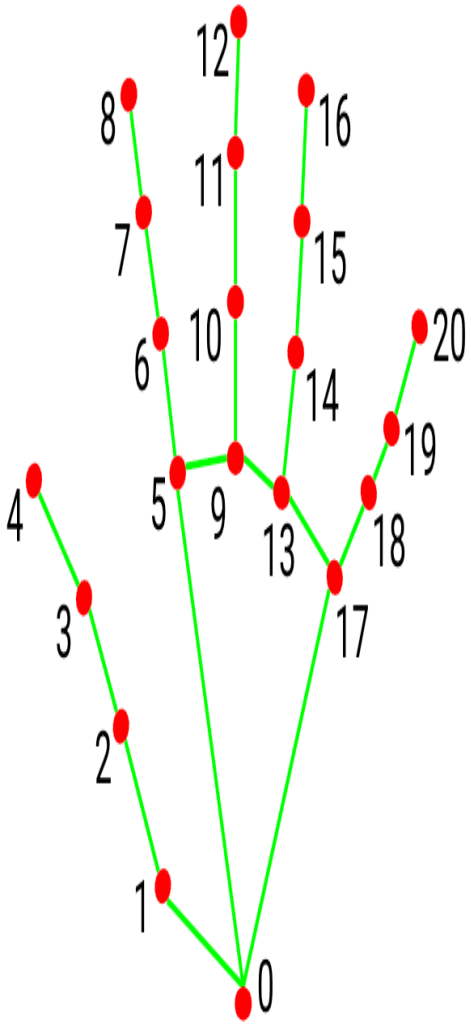
Hand Tracking using MediaPipe



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

1. Introduction

Hand tracking is a fundamental aspect of computer vision with applications in human-computer interaction, virtual reality. We used Python implementation of a hand tracking system using the MediaPipe

library. The objective is to detect and track hand movements, recognize gestures.

2. Methodology

2.1 Libraries Used

OpenCV (Open Source Computer Vision): Utilized for video capture and image processing. The cv2.VideoCapture class is employed to access the webcam feed, and various functions from OpenCV are used for image manipulation.

NumPy: Essential for numerical operations on image data. It facilitates efficient array manipulation, particularly in the context of image coordinate transformations.

MediaPipe: A powerful library that provides pre-trained models for various computer vision tasks, including hand tracking. The mp.solutions.hands module is employed for detecting and tracking hands, while mp.solutions.drawing_utils is used to visualize the hand landmarks and connections.

2.2 Hand Tracking Algorithm

2.2.1 Initialization

The script begins by importing necessary libraries and initializing the MediaPipe hands module. This includes creating instances of mp.solutions.drawing_utils and mp.solutions.hands.

2.2.2 Video Capture

A connection to the webcam is established using the cv2.VideoCapture class. This connection enables real-time access to video frames, forming the basis for subsequent image processing.

2.2.3 Hand Tracking Loop

The core of the algorithm involves a continuous loop, where each iteration processes a new frame from the webcam.

2.2.4 Image Preprocessing

The color space of the frame is converted from BGR to RGB, and the image is flipped horizontally. These preprocessing steps ensure compatibility with the requirements of the MediaPipe hands module.

2.2.5 Hand Detection

The hands.process function is applied to obtain hand detection results for the current frame. These results include the positions of hand landmarks and additional information.

2.2.6 Result Rendering

If hands are detected in the frame, the script utilizes mp_drawing.draw_landmarks to visualize the detected landmarks and hand connections on the image.

2.2.7 Gesture Recognition

The script analyzes the positions of specific landmarks, specifically the thumb and index finger tips, to determine whether the hand is open or closed.

2.2.8 Display and User Interaction

The processed image is displayed in a window titled 'Hand Tracking.' Users can exit the application by pressing the 'q' key.

3. Results

The system successfully detects and tracks hands in real-time, displaying landmarks and connections. Additionally, the application recognizes whether the hand is open or closed based on the relative positions of thumb and index finger tips. This functionality is visually represented on the displayed image.

4. Accuracy and Limitations

While the system demonstrates effective hand tracking, its accuracy may be influenced by factors such as lighting conditions and hand occlusions. Future enhancements could involve refining the model or incorporating additional computer vision techniques to mitigate these limitations.