SSRG International Journal of Electrical and Electronics Engineering
ISSN: 2348-8379/ https://doi.org/10.14445/23488379/IJEEE-V11I11P123

Volume 11 Issue 11, 231-240, November 2024
© 2024 Seventh Sense Research Group®

Original Article

Real-Time Full-Body Detection Using Computer Vision: Leveraging OpenCV and MediaPipe

Megha Chakole[1], Shrikant Sontakke[2], Lokesh Umredkar[3], Virendra Rathod[4], Roshan Umate[5], Sanjay Dorle[6]

[1,2,3,4]*Department of Electronics and Telecommunication Engineering, Yeshwantaro Chavan College of Engineering, Maharashtra, India.*
[5]*Department of Research and Development, Datta Meghe Institute of Higher Education and Research, Maharashtra, India.*
[6]*Department of Electronics Engineering, G.H. Raisoni College of Engineering, Maharashtra, India.*

[1]*Corresponding Author: mdorle@gmail.com*

Received: 13 September 2024 Revised: 14 October 2024 Accepted: 12 November 2024 Published: 30 November 2024

***Abstract** - This research includes a complete body detection system created using a computer vision library called OpenCV, which is primarily utilized to work on projects connected to image and video processing. Applications such as smart surveillance, human-machine interface, HMI, and human behavior all need body detection. This paper highlights the challenges that develops encounter while creating these kinds of applications. The challenges are choosing appropriate machine learning models and optimizing system performance. The primary goal of this work is to overcome the obstacles and identify solutions for them. OpenCV is one of the most potent and successful library computer vision tools, along with a few image-processing methods that yield real-time data on human movement and interaction. The focus of this research is a MediaPipe framework that Python developers commonly use to gather data on human interaction and real-time video capture. MediaPipe ensures that these programs run reliably and frees engineers to focus on improving algorithms. This research's finding offers potential human body detection approaches utilizing Mediapipe and OpenCV, which enable precise and comprehensive views of body posture, hand, and facial recognition.*

***Keywords** - Computer Vision, Full body detection, Image Processing, Mediapipe, OpenCV.*

1. Introduction

The technical world is seeing a rapid rise in artificial intelligence, which is having a significant impact on many different disciplines. The difficult and rapidly evolving subject of computer vision is used to analyze and gather precise environmental data. The actual and digital worlds must be brought closer together, which is made possible by software advancements and the ability to learn from recorded films and photos. The fundamental component of this subject is the OpenCV computer vision package, which enables and offers an appropriate environment for developing applications that can recognize and detect objects, including human body parts. Facial recognition and detection are crucial aspects of computer vision that are becoming more and more common in the domains of social media, education, and security. This study highlights OpenCV's contribution to face detection and recognition by showcasing its techniques and uses in Python-based systems [1]. Several components, including artificial intelligence, image processing, and computer vision, are merged into human body detection. It analyses and displays the posture of human body motions using digital images and videos. As a result, human-machine interaction may get better. These include robotics, medical analysis, behavioral biometrics, robotic surveillance systems, sports, entertainment, and gesture and posture recognition. Posture detection technology can help with medical diagnosis, boost sports training, increase safety, and provide more natural and efficient human-computer interactions by allowing computers to identify and analyze human motion [2]. As a crucial element of human-computer interaction, hand gesture recognition provides a natural means of communication with computers. Keyboards, mice, and, more recently, voice-to-text techniques have been the conventional means of operating computers. However, gestures offer an organic way to communicate that can be used to make technology engagement more natural. Some drawbacks of current hand gesture detection techniques include the need for hands to be placed in particular locations and the need for intricate coding. By identifying and counting every hand and finger visible to the camera, regardless of position, the suggested approach, in contrast, gets around these difficulties. The model, developed in Python, is understandable and effective, making it useful for developers and users. Gesture recognition is much easier with this method [3]. In the Philippines, in particular, closed-circuit television, or CCTV, is a commonly utilized surveillance technique used for security and monitoring by both government and private organizations. Due to the popularity of CCTV due to its cost and efficacy, legislation such as the CCTV Cameras for Security Act of 2012 was passed, requiring the installation of CCTV cameras in business enterprises as a means of deterring crime.

Digital Video Recorders (DVRs) and CCTV systems are connected. These DVRs have alert features that allow them to record motion, human movement, and object movement. They then save all of this data in the designated data storage area. Thanks to technological advancements, CCTVs are regularly utilized as surveillance systems on PCs and smartphones [4]. Another crucial component in many fields is object detection, which is accomplished via computer vision. Notifying someone and finding the thing after it is detected is also crucial. OpenCV is used with Python. Strong instrumentation for object detection is provided after pairing and importing OpenCV into Python. OpenCV offers a plethora of features and techniques, including image and video processing, selecting detection algorithms, gathering and presenting data, and more. This is the most effective way for machines to locate objects because it can produce accurate real-time data. Technology advancements have raised the need for safety and security, which has led to a growth in the number of surveillance systems, such as digital cameras. It can take a long time and require much work to monitor and store the surveillance data. Automated systems that can monitor and identify moving items are being developed as a solution to this problem.

These systems, whether analog or digital, rely on cameras to monitor areas and detect movement. Motion detection aims to automatically identify and record movement within a specified area, improving the efficiency and effectiveness of surveillance and security measures [5, 6].

2. Related Work

Numerous methods for identifying moving objects have been developed recently. Statistical background modelling is one method that uses edge comparison between the current video frame and the backdrop to detect moving objects. This approach, however, has trouble when moving items cross over into the margins of the background. You Only Look Once (YOLO) rapidly predicts bounding boxes and class probabilities by studying the full image. This technique has several advantages. Another study looked at utilizing OpenCV on Android and included a beginner's guide as well as examples of the impacts of several image filters, such as colour conversion and grayscale. When a webcam's image is analysed by a motion detection system, it may identify movement and, if needed, sound an alert at vital areas like banks. Using video data, a Gaussian Mixture Model (GMM) based foreground-background motion detection method extracts moving objects from the background. This system applies filters to minimize noise from light, shadows, and wind and optimizes the GMM to shorten processing times. Particle filter strategies for tracking objects have been investigated in other studies. These techniques involve comparing template forms and estimating regions in video scenes or utilizing edge information and temporal difference detectors to improve object recognition accuracy. All of these investigations show how far moving object tracking and detection have come, each with its own advantages and disadvantages [7, 8].

The three primary processes in face recognition are feature extraction, face recognition, and face detection. Face detection first finds and concentrates on faces inside pictures or videos. This technique separates the face from the background and other objects; however, it may not work well for faces that overlap or have similar facial features. Face recognition then establishes the identity of the face by comparing it to previously identified faces. To identify a person, the system considers a number of facial characteristics, including skin tone, structure, and form. Lastly, the process of face extraction records and extracts particular traits from the identified face. Even though these systems are meant to recognize and authenticate faces, they may have trouble processing faces that are strikingly the same or overlapped. Techniques like principal component analysis with distance are used to increase accuracy [9-11]. Recent developments in facial recognition technology have made numerous noteworthy techniques and systems possible. One method, FaceNet, learns to map photographs of faces to a space where the distances between faces accurately reflect their similarity. While Local Binary Patterns (LBP) encode local pixel patterns to accommodate illumination variations, other approaches, such as Eigenfaces, use Principal Component Analysis (PCA) to capture significant facial changes. Linear Discriminant Analysis (LDA), often known as Fisher faces, focuses on differentiating features between faces [12]. In recognition tests, Deep Convolutional Neural Networks (CNNs) have demonstrated great success rates in extracting intricate facial patterns. Furthermore, 3D face recognition recognizes faces precisely, even when their position or expression changes, by using depth information from specialized sensors. By increasing accuracy, addressing a range of situations, and offering reliable solutions, these methods help to improve face recognition systems [13].

Recent work in media analysis has concentrated on various approaches to audio and video data processing and comprehension. The usual process for analyzing media files involves separating them into audio and visual streams [14-16]. Each input generates a single output in directed graphs, enabling efficient computation in popular neural network frameworks like TensorFlow and PyTorch. By handling more intricate and dynamic behaviors where inputs might result in various outputs, MediaPipe functions at a higher level [17, 18]. For the analysis of media with complex semantics, this makes it very useful. For real-time media processing, other frameworks like Beam and Dataflow are more suited because they handle massive amounts of data in chunks. Rather than concentrating on useful media analysis, Ptolemy models concurrent systems. While MediaPipe avoids the complexities of inter-process communication, ROS leverages graph-based processing in robotics. OpenCV's Graph API is restricted to image processing, whereas GStreamer is utilized for media editing. MediaPipe is more appropriate for audio and sensor data processing due to its adaptability and streaming data capability [19, 20].

3. Materials and Methods

This work demonstrated the real-time detection and drawing of landmarks for hands, faces, and body positions using MediaPipe's Holistic model while recording webcam footage. The first step is to establish a connection to the webcam and configure MediaPipe's Holistic model with a minimum of 0.5 tracking confidence and 0.5 detection confidence. Since MediaPipe requires RGB input, each frame of the webcam stream is transformed from Blue, Green, and Red (BGR) to Red, Green, and Blue (RGB) color space as it is being recorded frame by frame.

After that, each frame is processed by the Holistic model to identify different landmarks, such as hand and body poses and facial features. The drawing tool used to identify landmarks on objects, such as human body parts, is called Mediapipe. This framework allows for the identification of body parts and the drawing of landmarks on the computer screen. These landmarks feature colorful patterns as well as unique facial and hand designs. When landmarks are found, the picture is transformed into BGR color space so that it may be seen on the display screen. This system utilizes the webcam to identify various body parts, including the face and hands, as well as fingers and palms. It analyzes posture and motion using the Media Pipe holistic model. Until the user closes the open application, this system continues to function. It gives a simple way to identify body sections. Figure 1 illustrates steps for full body detection using Open CV. The processing steps are explained as follows.

1. The program gathers input video feeds using the Web Camera.
2. Frame Acquisition: Every frame from the webcam feed is captured.
3. Color Conversion: The frame is transformed from BGR to RGB color space as needed by the MediaPipe library.
4. MediaPipe Holistic Model: The converted RGB frame is used to simultaneously detect face, position, and hand landmarks.
5. Face Landmarks Detection: Detects points around face features
6. Detection of hand landmarks: Identifies both right and left hands
7. Landmark Detection: Identifies essential body joints for pose
8. Visualization: OpenCV detects landmarks and overlays them onto the original frame.
9. Display: An annotated frame with detected landmarks is displayed in a window.
10. User Interaction: The system will continue processing frames until the user presses the 'q' key to quit.
11. Cleanup: When the user exits, the webcam feed is released and all OpenCV windows close.

The following methods are required to develop an algorithm to detect the full body.

3.1. Image Processing

The image needs to be converted from BGR to RGB because of the framework used. Mediapipe models work on RGB images. The mathematical equation for conversion can be given by:

$$Irgb = cv2.cvtColor(Ibgr,cv2.color\_bgr2rgb) \quad (1)$$

Where, Irgb and Ibgr are the RGB and BGR image matrices, respectively.

3.2. Landmark Representation

When landmarks are detected, they are represented in the form of 2D or 3D Coordinates, depending on the received output and the system compatibility. The 2D landmark is mathematically given as:

$$Lf = \{(Ai,Bi) \mid I = 1,2,\ldots..,Nf\} \quad (2)$$

Where:

- Lf is the set of 2D facial landmarks
- Ai and Bi are the co-ordinates of i^{th} landmark.
- Nf is the total number of facial landmarks.

Similarly, hand and pose landmarks are given using the same mathematical equation, i.e.,

$$Lf = \{(Ai,Bi) \mid I = 1,2,\ldots..,Nf\} \quad (3)$$

3.3. Drawing Landmarks

The landmarks generated are displayed on the image using some drawing functions according to their coordinates. The drawing function mathematically can be given as:

$$D (L, C, T) = cv2.circle\{I, (a, b), R, C, T\} \quad (4)$$

Where,

- D is the function for drawing landmarks.
- L = (a, b) the coordinates of the landmark.
- C is color used for drawing
- T is the thickness of the drawing
- R is the radius of the circle for the landmark

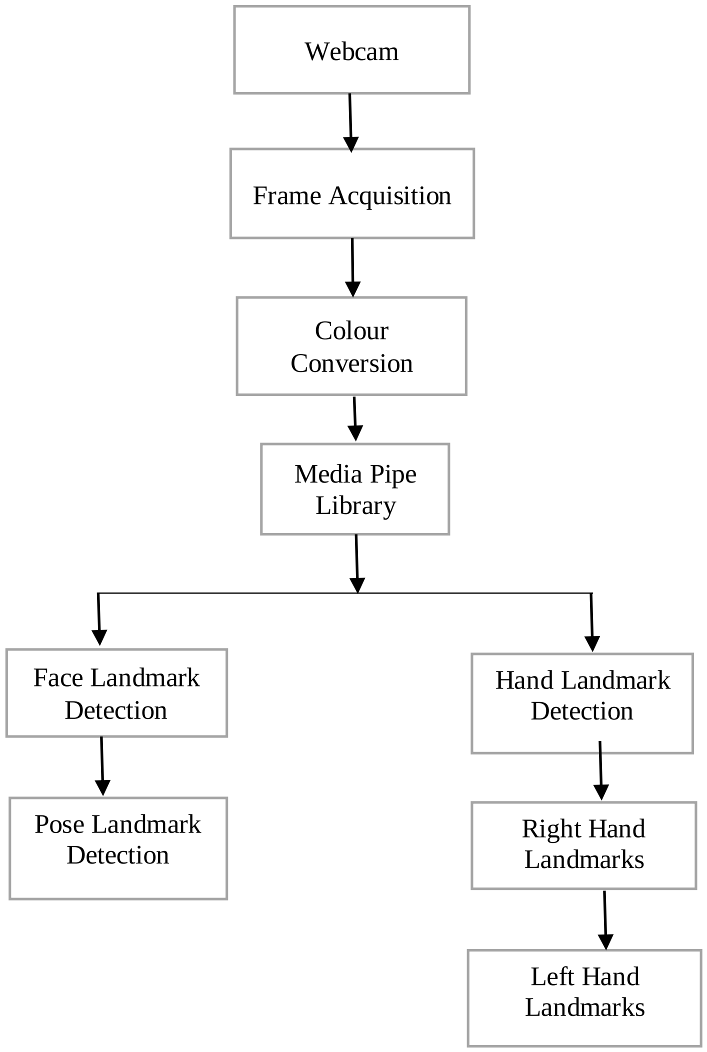

Fig. 1 Steps for full body detection using OpenCV

3.4. Threshold Confidence

The confidence thresholds are used to decide whether to supply these landmarks.

$$Cd, Ct >= 0.5$$

Where,

- Cd is the minimum detection confidence
- Ct is the minimum tracking confidence

The flowchart for the algorithm to detect the full body is shown in Figure 2.

3.5. Media Pipe Model

Face Mesh: It offers a detailed mapping of 468 face landmarks, making facial recognition extremely accurate. It generates 3D points across the face effectively for real-time applications using a lightweight regression model.

Hand tracking: This feature allows for a thorough examination of hand movements by detecting and tracking 21 distinct landmarks on each hand. Accurate hand tracking, even for moving hands, combines a localization model with a palm detector.

Stance estimation: It identifies 33 important body joints, including the knees and shoulders, to capture a full-body stance structure. For pose estimation, it uses a two-phase detection approach to determine a Region of Interest (ROI) for the pose before employing regression to recover exact landmarks.

Figure 2 illustrates how to detect bodies using the provided Python code, which makes use of OpenCV and the MediaPipe module. Below is a full description of each step in the flowchart:

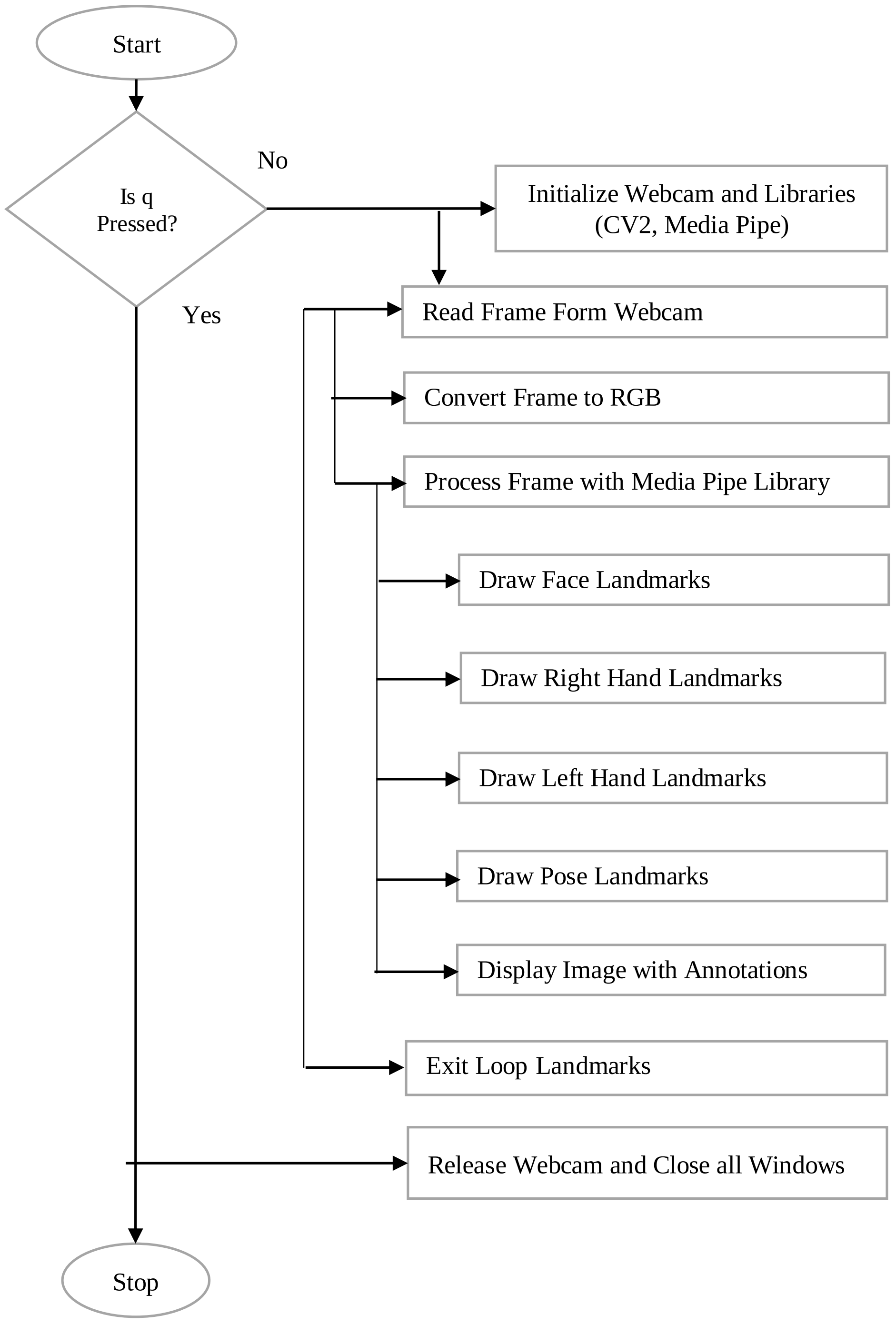
Start
Is q Pressed?
No
Yes
Initialize Webcam and Libraries (CV2, Media Pipe)
Read Frame Form Webcam
Convert Frame to RGB
Process Frame with Media Pipe Library
Draw Face Landmarks
Draw Right Hand Landmarks
Draw Left Hand Landmarks
Draw Pose Landmarks
Display Image with Annotations
Exit Loop Landmarks
Release Webcam and Close all Windows
Stop

Fig. 2 Flowchart for the algorithm used to detect full-body

1. Import libraries like cv2 (OpenCV) and mediapipe, and initialize the webcam for video input.
2. Loop while webcam is open: The code will process frames from the webcam till it is closed.
3. Each iteration of the loop involves reading a frame from the webcam feed.
4. Convert frame to RGB: The frame's color space is converted from BGR (recorded by OpenCV) to RGB for Media Pipe processing.
5. Process the frame with the MediaPipe Holistic model to detect facial, position, and hand landmarks.
6. Identify and draw face landmarks on the frame based on the drawing parameters.
7. Draw right-hand landmarks on the frame.
8. Draw left-hand landmarks on the frame.
9. Draw position landmarks to illustrate important body joints on the frame

10. Display image with annotations: The "HOLISTIC MODEL DETECTION" window displays the annotated frame with all recognized landmarks
11. The code looks for the 'q' key to exit the loop and end the program.
12. To release the webcam and close all windows, press 'q'. When the 'q' key is pressed, the webcam feed is released, and all OpenCV windows are closed, thereby terminating the program.

The process of full body detection using OpenCV, taking into account inputs from the web camera and body landmarks and displaying the result, is depicted in the aforementioned flow chart.

The following are the strengths of the model.

- Accuracy: Ensures thorough body part detection even in complicated positions by providing exact landmark localization.
- Efficiency: It can function easily on the majority of CPUs without the need for a specialized GPU because it is optimised for real-time performance.
- Integration: Simplifies the implementation process by combining facial, hand, and body position detections in a single pipeline.

Following are the limitations of the algorithm

- 2D Landmark Detection: Without extra hardware, genuine depth perception may be imprecise even though it produces normalized 3D coordinates.
- Environmental Dependency: The quality of detection may be impacted by performance deteriorating in dimly illuminated areas or from extreme camera angles.
- Occlusions: When body portions are hidden, the model may have trouble and its accuracy may be limited.

4. Results and Discussion

OpenCV and Mediapipe are utilized in this paper, together with real-time holistic model detection that incorporates face, hand, and position landmarks to record webcam footage.

Capturing Video from WebCam: The main purpose of WebCam is to record video from either an external USB camera linked to a computer or the laptop's built-in default camera.

Initializing MediaPipe Holistic Model: The Mediapipe holistic model can identify hands, the face, and position landmarks, among other body parts. The initialization of this model is to gather the current landmarks.

Processing Each Frame: Video frames are transformed from BGR to RGB since MediaPipe uses RGB image formats for processing. A holistic model is processed to identify the landmarks for the hand, face, and posture. Following this procedure, OpenCV is used to transform the video frame picture back to BGR for display.

Displaying the Result: After all, the computer or laptop's screen displays a fully detailed view of body detection together with precisely painted landmarks.

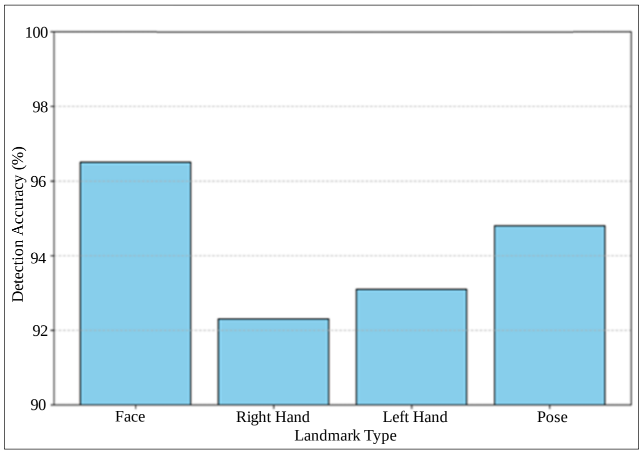

Fig. 3 Detection accuracy by landmark type

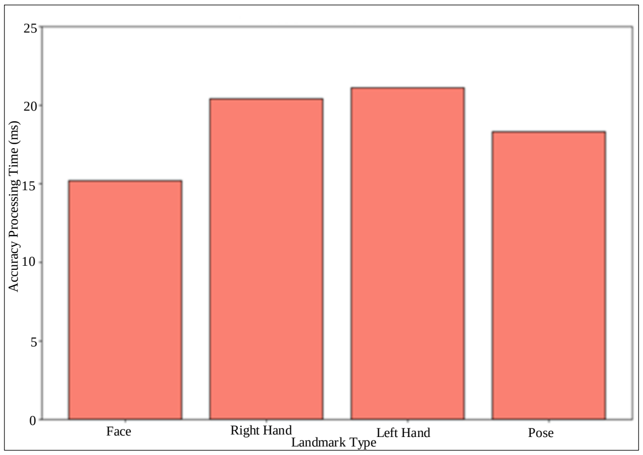

Fig. 4 Average processing time by landmark type

Figure 3 displays the various landmark kinds' detection accuracy. Facial landmarks have the highest accuracy, at over 96%, according to the graph. The accuracy of the right and left hands is slightly slower, at about 92%, with stance detection falling in between.

The average processing time for each sort of landmark is contrasted in Figure 4. Face detection is the fastest (around 15 ms), but hand landmarks (left and right) take the longest (more than 20 ms apiece), with stance detection being a little quicker.

YOLO, OpenPose and Mediapipe are the three systems that are benchmarked depicted in Figure 5. Mediapipe offers the highest detection accuracy (95%) and the fastest processing time (16 ms). OpenPose and YOLO, on the other hand, sacrifice accuracy for lengthier processing periods, with respective speeds of 92%, 23.5 ms, and 93.5%, 30.2 ms. Comparing the OpenPose and YOLO, Mediapipe performs well.

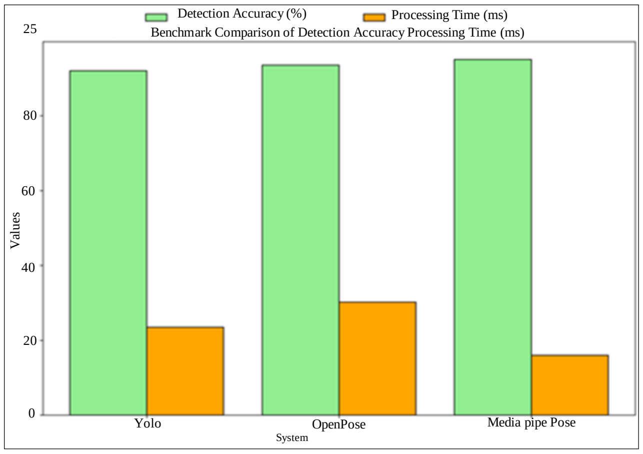

Fig. 5 Comparison of detection accuracy and processing time for various systems

The research's findings on full body detection with computer vision lead to improved data accuracy and the ability to run an application utilizing reference studies; this is a summary of the project's testing and outcomes:

1. The ground truth annotations and the actual landmarks produced by the screening are compared to examine the entire body. For this, the entire body's real-time image is employed.
2. System performance is also assessed under specific circumstances. Cross-checking was also done in various scenarios. Difficulties like congested areas and numerous faces are also utilized to assess efficacy.
3. The system's resources, such as processing speed and working time, are assessed. This also covers the system's memory use and processing time for different hardware applications.
4. System evaluation based on data collected in real time. It also includes the entire region required for body detection. These domains are comparable to sports, academic fields, etc.
5. The entire body detection system is contrasted with other systems that are referenced and utilized for item movement detection and facial, head, or hand detection.

Other authors' and developers' findings indicate that system testing has been completed. The effectiveness and capacity to detect various bodily sections and the total body are assessed. The following images show the real-time testing of the system:

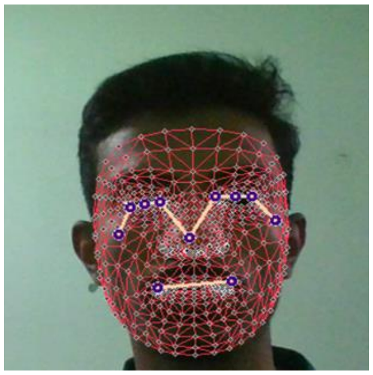

Fig. 6 Face detection using OpenCV in real-time

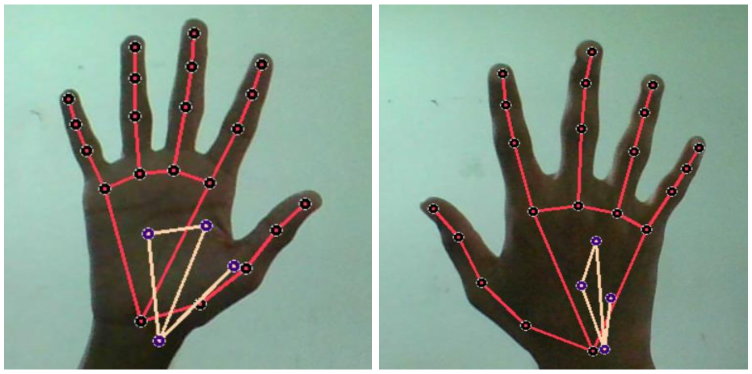

Fig. 7 Right and left-hand palm and finger landmarks detection using MediaPipe

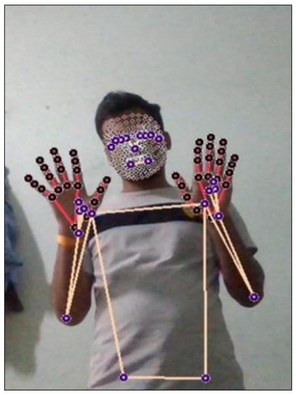

Fig. 8 landmarks connected by line segments are used to identify the hands

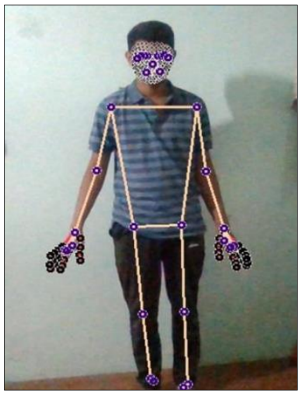

Fig. 9 landmarks connected by line segments are used to identify the face and body attitude

An example of face detection using OpenCV in real time is shown in Figure 6. This face's emphasized and finely developed features are easily visible in this photo. This visual aid displays the relationships and landmarks colored-drawn on the face. Important hand landmarks, like the palm and fingers, are identified and connected by the MediaPipe model, as illustrated in Figure 7. The lines and circles, which have been given different colors and thicknesses to improve visual clarity, depict the locations and connections that have been identified.

This illustration aids in the understanding of hand gestures and movements. Successful detections are shown in Figures 8 and 9, where landmarks connected by line segments are used to identify the hands, face, and body attitude. Through the use of a tessellation pattern on the face mesh and accompanying hand and body landmarks that describe their structural makeup, real-time facial expressions and human movement are portrayed in depth. This research provides the following adjustments and fixes for some of the references' limitations.

Table 1. Analysis of references versus research contributions

Findings of References	Research Contribution of the Manuscript
The majority of research investigations have focused on either head, hand, motion, or face detection.	All significant body components, such as the face, hands, fingers, and legs, are fully detected in this study.
For the goal of detection, the majority of the studies from the references used C, C++, and other languages.	This work demonstrates the usage of Python programming and, more importantly, image processing, machine learning, and libraries like Mediapipe and OpenCV.
It is impossible to recognize or difficult to notice multiple faces at once.	The work presented in this publication allows multiple-face detection to be used.
	In contrast to the references, this study used RGB to detail the landmarks of the hands and the face.

The overall results of this research are performance, applications, and real-time landmark identification. Facial landmarks, hand landmarks, and body pose landmarks are shown on the screen in real-time landmark detection. The different body parts are properly visualized using the MediaPipe model. Numerous other sectors where human body sensing is necessary, such as gesture recognition, fitness monitoring, animation, and many more, can benefit from this research. Because the Mediapipe Holistic model operates in real-time, the system's processing power and the caliber of the WebCam attached to it will determine how well the model performs. The characteristics of the computer vision framework and the potential applications of complete body detection are the main focus of this paper.

The efficiency of this work improved and publishers and researchers should leverage the MediaPipe framework to create more sophisticated pipelines. This aids in algorithm optimization, written code efficiency, and code length reduction using necessary libraries and extensive Python library support. Moreover, more intricate body positions can be identified. It is important to record various positions and motions with greater depth. In order to increase accuracy, robustness, and practicality in real-world situations, future research in computer vision-based complete body identification may concentrate on refining existing techniques, overcoming technological problems, and investigating novel approaches.

4.1. Data Privacy

Real-time body detection and tracking systems must provide both data privacy and ethical compliance, particularly in surveillance settings. This research preserves user anonymity while managing privacy through anonymisation approaches by limiting the collection of identifying data to body motions and gestures rather than specific facial features.

Furthermore, encryption and restricted access procedures are in place to prevent unauthorised handling and secure storage methods are used to safeguard any stored data. Prioritising user consent, being transparent about the methods and purposes of data collection and retention, adhering to data protection laws, and lowering the risks of unforeseen secondary uses like commercial apps or unnecessary analytics are all ways to resolve ethical concerns. The system is ethically built to prevent unwanted profiling and limit access to data to authorised personnel only. Particularly when employed in fields like healthcare, education, or security, this methodical approach centred on anonymisation, secure storage, explicit user consent, and stringent data handling protocols helps handle the complex privacy and ethical challenges raised by real-time monitoring technologies.

4.2. Challenges

Economic and operational concerns that impact the validity and reliability of human detection are the primary challenges in its development. High accuracy is challenging to attain in complex scenarios where low-level objects or backdrops, shifting stances, and occlusion can all make identification more difficult. Another challenge is operating equipment with limited computing because interaction depends on measurement speed and precision. Another concern is scalability, particularly in big deployments where the system must handle massive volumes of data. It is difficult and frequently constrained by the variety of data available to ensure that the model generalizes well across various situations without requiring retraining. Integration with current systems, like mobile phones or Internet of Things platforms, is also necessary for collaboration. It is challenging to develop effective solutions for people due to hardware constraints, particularly with low-power devices like drones or cameras, as well as the requirement for high-quality instructional materials and recordings.

5. Conclusion

This paper presents the effective use of Media Pipe architecture and OpenCV to a whole-body detection system. It is a real-time body posture detection system that aims to track and gather motion data related to the movements and postures of the human hands, face, and overall body.

This approach takes advantage of the most effective and popular framework, MediaPipe. Along with computer vision's capacity for image processing, it is related to the discovery of solutions to typical issues like low light detection and multi-face photos. This system operates without interruption or deviation unless the user grants permission. It is utilized to solve safety and security issues in healthcare, educational institutions, and smart surveillance. Python is the programming language utilized, which offers a user-friendly coding environment in addition to extensive library support. This study also contributes to developing and improving computer vision technology by offering simple fixes for common issues with surveillance and body detection. It encourages the use of software technologies and inspires programmers and developers to create more of these kinds of applications for the advancement of technology and society.

5.1. Future Work

Future developments in the human detecting system may focus on a few key topics. Using deep learning models like Vision Transformers (ViTs) or Convolutional Neural Networks (CNNs), which increase accuracy in complex, crowded scenarios with occlusions, fluctuating illumination, and body angles, could improve human location estimation and segmentation. Enhancements to object tracking methods like Kalman Filters or optical flow will be necessary for real-time gesture identification and tracking, potentially expanding their applications in interactive settings, healthcare, and security. Cross-platform optimization guarantees real-time performance and reduces the computational load for wearables and smartphones. LiDAR and infrared cameras are examples of multimodal sensors that can be utilized to enhance detection in low light. Finally, improving scalability to accommodate numerous users in crowded or huge settings would need to look at cloud-based options for centralized processing and improving computer efficiency.

References

[1] Ramadan TH. Hasan, and Amira Bibo Sallow, "Face Detection and Recognition Using OpenCV," *Journal of Soft Computing and Data Mining*, vol. 2, no. 1, pp. 86-97, 2021. [Google Scholar]

[2] Anand Upadhyay et al., "Body Posture Detection Using Computer Vision," *SSRG International Journal of VLSI & Signal Processing*, vol. 7, no. 1, pp. 6-10, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[3] Satyavati Jaga, "Hand Detection and Tracking Using OpenCV and Python" *International Journal of Advanced Research in Engineering and Technology*, vol. 11, no. 12, pp. 3261-3266, 2020. [Publisher Link]

[4] Gretchel Karen L. Alcantara et al., "Head Detection and Tracking Using OpenCV," *2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM),* Baguio City, Philippines, pp. 1-5, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[5] P. Nikitha et al., "Object Detection Using OpenCV With Python," *International Research Journal of Modernization in Engineering Technology and Science*, vol. 5, no. 6, pp. 762-767, 2023. [CrossRef] [Publisher Link]

[6] Shubham Mishra et al., "An Intelligent Motion Detection Using OpenCV," *International Journal of Scientific Research in Science, Engineering and Technology*, vol. 9, no. 2, pp. 51-63, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[7] Kruti Goyal, Kartikeya Agarawal, and Rishi Kumar, "Face Detection and Tracking Using OpenCV," *2017 International Conference of Electronics, Communication and Aerospace Technology*, Coimbatore, India, pp. 474-478, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[8] Arun Binoy et al., "Face Recognition System Using OpenCV," *International Research Journal of Engineering and Technology*, vol. 10, no. 8, pp. 128-132, 2023. [Publisher Link]

[9] Camillo Lugaresi et al., "MediaPipe: A Framework for Building Perception Pipelines," *arXiv Preprint*, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[10] Dhammapal Suradkar, Ashish Jogad, and Pramod Talole, "Face Detection Using OpenCV," *International Journal of Advanced Research in Science, Communication and Technology*, vol. 7, no. 1, pp. 18-27, 2021. [Publisher Link]

[11] Kaleem Ullah et al., "Comparison of Person Tracking Algorithms Using Overhead View Implemented in OpenCV," *2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON)*, Jaipur, India, pp. 284-289, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[12] Nidaa Falih Hassan, Introduction to Computer Vision and Image Processing, 2016. [Online]. Available: http://www.uotechnology.edu.iq/dep-cs/mypdf/subjects/4sw/4ip.pdf

[13] Hongping Cai et al., "The Crossdepiction Problem: Computer Vision Algorithms for Recognising Objects in Artwork and in Photographs," *arXiv Preprint*, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[14] Gudala Lavanya, and Sagar Dhanraj Pande, "Enhancing Real-time Object Detection with YOLO Algorithm," *EAI Endorsed Transactionson Internet of Things*, vol. 10, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[15] Weijia Zhang et al., "Combined MediaPipe and YOLOv5 Range of Motion Assessment System for Spinal Diseases and Frozen Shoulder," *Scientific Reports*, vol. 14, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[16] Jose Sigut et al., "OpenCV Basics: A Mobile Application to Support the Teaching of Computer Vision Concepts," *IEEE Transactions on Education*, vol. 63, no. 4, pp. 328-335, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[17] Vijay Kumar Sharma, "Designing of Face Recognition System," *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, Madurai, India, pp. 459-461, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[18] B. Thaman, T. Cao, and N. Caporusso, "Face Mask Detection Using MediaPipe Facemesh," *2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*, Opatija, Croatia, pp. 378-382, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[19] Abhinav Prajapati, Rahul Chauahan, and Himadri Vaidya, "Human Exercise Posture Detection Using MediaPipe and Machine Learning," *2023 3rd International Conference on Advancement in Electronics & Communication Engineering (AECE)*, Ghaziabad, India, pp. 790-795, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[20] Yaseen et al., "Next-Gen Dynamic Hand Gesture Recognition: MediaPipe, Inception-v3 and LSTM-Based Enhanced Deep Learning Model," *Electronics*, vol. 13, no. 16, 2024. [CrossRef] [Google Scholar] [Publisher Link]