HUMAN POSE ESTIMATIONION USING MACHINE LEARNING

A Project Report

submitted in partial fulfillment of the requirements

of

AICTE Internship on AI: Transformative Learning
with
TechSaksham – A joint CSR initiative of Microsoft & SAP

by

S. ARCHANA , Cybernetic.arch005@gmail.com

Under the Guidance of

P.RAJA

ACKNOWLEDGEMENT

I would like to take this opportunity to express our deep sense of gratitude to all individuals who have supported us directly or indirectly during this internship project.

Firstly, we would like to thank our mentor, **P. Raja Sir**, for being an exceptional guide and mentor throughout this journey. His valuable teachings on OpenCV, constructive feedback, and constant encouragement were the driving forces behind the successful completion of this project. The confidence he instilled in us served as a significant source of motivation. It has been an honor to learn under his guidance. His insightful lessons and guidance not only aided in project execution but also contributed to our professional growth.

I would also like to extend our heartfelt thanks to **Pavan Sumohana SIR** for effectively managing the internship program and providing prompt assistance with all LMS-related queries. His dedication and support ensured a smooth learning experience, allowing us to focus entirely on our work.

This internship has been an enriching experience, and we are deeply grateful for the mentorship and support provided by both of them.

Abstract

The "**Pose Detection Using Mediapipe and Streamlit**" project integrates Mediapipe's pose estimation capabilities with Streamlit's user-friendly interface to create a real-time application for detecting human body pose landmarks in images and videos. This project serves as an accessible solution for analyzing body movements in sports, fitness, and rehabilitation applications, making it suitable for academic or internship purposes.

Problem Statement

Detecting human body poses and movements is a critical need in fields such as sports performance analysis, physical therapy, and motion tracking. However, existing pose detection systems often require complex setups or expensive hardware, limiting accessibility for learners and professionals.

Objectives

- Develop a simple, interactive application for pose detection using Mediapipe.

- Provide support for both image and video uploads for flexibility in input formats.

- Design an intuitive interface using Streamlit, enabling easy use for beginners and professionals.

- Deliver a lightweight, cost-effective solution suitable for academic and internship environments.

Methodology

The application employs Mediapipe's Pose module to detect body landmarks, such as joints, in uploaded media. Streamlit is used to build a web-based interface where users can upload images or videos for pose detection. Input media is processed frame-by-frame, and pose landmarks are visualized with clear connections. Results are displayed directly within the web application, providing instant feedback.

Key Results

The application successfully detects and visualizes body pose landmarks in both static images and dynamic video frames. Its simplicity and effectiveness make it a reliable tool for understanding pose detection concepts without requiring specialized hardware.

Conclusion

This project demonstrates the feasibility of creating a beginner-friendly, real-time pose detection tool. By combining Mediapipe's advanced pose estimation with Streamlit's ease of use, it offers a practical solution for applications in sports and health-related domains while serving as a foundational project for AI-DS learners.

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 Problem Statement:

The detection of human body poses and movements is an essential task in various fields such as sports performance analysis, fitness monitoring, physical therapy, and rehabilitation. Accurate pose detection helps in understanding body movements, preventing injuries, and improving performance. However, most available pose detection systems rely on complex algorithms, specialized hardware, or high computational power, making them inaccessible for everyday users, students, and professionals without advanced resources. Furthermore, the lack of real-time feedback in existing tools hinders the interactive learning and immediate application of pose analysis. This project aims to address these issues by developing an accessible, real-time, and lightweight pose detection system using Mediapipe and Streamlit.

1.2 Motivation

The motivation for choosing this project stems from the growing need for affordable and accessible pose detection solutions in various sectors, such as sports, health, and education. In sports, accurate pose analysis allows athletes to fine-tune their movements, enhance performance, and avoid injuries. In physical therapy, it assists therapists in monitoring patients' recovery by tracking their movements over time. Furthermore, this technology can be applied in fitness apps, virtual workouts, and even remote education for fitness instructors.

The potential impact of this project lies in its ability to democratize access to pose detection technology. By utilizing freely available libraries like Mediapipe and Streamlit, the project offers a scalable, easy-to-use solution that can be integrated into various applications. This project not only serves as a tool for practical applications but also acts as an educational platform for beginners and professionals looking to explore pose detection in AI and computer vision fields.

1.3 Objective:

The primary objectives of the **"Pose Detection Using Mediapipe and Streamlit"** project are:

1. **Develop a Real-Time Pose Detection Application**:

Create an easy-to-use application that utilizes Mediapipe's Pose Estimation model to detect human body landmarks (joints) in images and videos.

2. Support Multiple Input Formats:

Enable the application to process both static images and dynamic video files, providing flexibility for users to upload and analyze various types of media.

3. **Provide Interactive User Interface**:

Build an intuitive and responsive interface using Streamlit, allowing users to upload media and receive immediate visual feedback on detected poses with clear landmark connections.

4. Offer a Lightweight Solution:

Design the application to be lightweight, accessible, and user-friendly, without requiring high computational power or complex hardware, making it suitable for educational, research, and practical use in various domains.

5. Educational Experience:

Create a simple yet effective learning tool for students, researchers, and professionals to understand and apply pose detection techniques in real-time scenarios, supporting fields like sports performance analysis, fitness tracking, and rehabilitation.

1.4 Scope of the Project:

The scope of the "Pose Detection Using Mediapipe and Streamlit" project encompasses the development of a real-time pose estimation tool that can detect and visualize human body landmarks in both static images and video files. The project is designed to be accessible, lightweight, and interactive, allowing users to upload images or videos, view pose landmarks, and track movements with clear, visual feedback.

The key areas covered by the project include:

1. Pose Detection:

Utilizing Mediapipe's Pose Estimation model to accurately detect and display key body landmarks (such as joints, hands, and face) in real-time from images and video files.

2. Flexible Input:

Supporting both images and video uploads, making the application versatile for various use cases, including sports analysis, fitness tracking, and educational purposes.

3. Real-Time Processing:

Providing instant feedback on uploaded media, with visual representation of pose landmarks overlaid on the image or video.

4. User-Friendly Interface:

Designing a web-based interface using Streamlit that is easy to navigate and understand for users with limited technical knowledge.

5.Potential Applications:

Enabling use in fields such as sports performance analysis, rehabilitation therapy, virtual fitness sessions, and educational tools for learning pose detection and computer vision concepts.

1.5 Limitations of the Project

While the project successfully addresses several aspects of pose detection, it has certain limitations:

1.Pose Accuracy:

The accuracy of pose detection depends on the quality of the input media. Blurry, low-resolution, or occluded images/videos may result in less accurate pose estimations.

2. Limited to Human Pose Detection:

The application is designed specifically for human body pose detection and does not support the detection of other objects or animals.

3. Real-Time Limitations:

While the project supports real-time video processing, the performance may vary based on the hardware specifications of the device running the application, particularly for longer or higher-resolution videos.

4. No Advanced Features:

The project does not implement advanced features like multi-person detection, motion tracking, or 3D pose estimation, which could be useful in more complex applications.

5. Minimal Customization:

The application is designed for general use, and users may have limited options for customizing the detection parameters or adding additional functionalities beyond the basic pose detection features.

CHAPTER 2

Literature Survey

2. Literature Survey

A literature survey provides an overview of existing research, projects, and tools related to your project. It helps in understanding the current state of the field and identifies gaps that your project aims to address. Below is a structure for the literature survey:

2.1 Introduction to Pose Estimation

Pose estimation is a computer vision task that involves detecting and tracking human body poses from images or video frames. It is a crucial technology for various applications, including human-computer interaction, sports analytics, fitness tracking, and rehabilitation. Traditional pose estimation techniques used to rely heavily on complex hardware systems and manual feature extraction. However, with the advancements in machine learning and deep learning, pose estimation has become more accessible and efficient.

Mediapipe, a cross-platform framework developed by Google, has become a popular solution for real-time pose estimation. It uses machine learning models to detect and track human poses with remarkable accuracy, even in challenging real-world environments. The rise of frameworks like **Streamlit** has enabled developers to create simple web applications that integrate these advanced algorithms with minimal code.

2.2 Related Work on Pose Estimation

1. OpenPose (2018)

OpenPose, developed by the Carnegie Mellon Perceptual Computing Lab, was one of the first real-time multi-person keypoint detection systems. It works by detecting and mapping body parts, hands, facial landmarks, and even the pose of the upper body. OpenPose has been widely used in several fields such as virtual reality, fitness apps, and human-computer interaction. However, OpenPose requires substantial computational resources and is relatively complex to set up.

2. Mediapipe Pose (2020)

Mediapipe Pose, developed by Google, is a more efficient and lightweight solution for pose estimation. It uses a pre-trained machine learning model to detect and track body landmarks in real-time with high accuracy. The framework is suitable for mobile devices and can be integrated easily into various applications, making it a good choice for this project. Mediapipe has been used in numerous applications, including fitness tracking, gesture recognition, and augmented reality.

3. DeepPose (2014)

DeepPose, introduced by researchers at Google, was one of the first methods to use deep learning for pose estimation. It used convolutional neural networks (CNNs) for pose regression, achieving impressive accuracy on challenging datasets. While DeepPose laid the foundation for modern pose estimation, it relied on large datasets and computational power, which made it less suitable for real-time applications in resource-constrained environments.

4. PoseNet (2016)

PoseNet, developed by Google, is another deep learning-based pose estimation method that works well on mobile devices. It uses a single CNN to estimate the body keypoints in images. PoseNet is efficient and lightweight compared to OpenPose, making it ideal for real-time applications in mobile devices. Its use in pose estimation for human movement has been explored in fitness apps and AR environments.

2.3 Related Work on Real-Time Pose Detection with Streamlit

Streamlit, a web application framework for building machine learning and data science applications, has gained popularity for its ease of use and rapid deployment capabilities. There are several open-source projects that combine Streamlit with pose estimation algorithms like Mediapipe to create real-time applications.

1. Streamlit + Mediapipe for Pose Estimation

Several community-based projects have demonstrated how to use Streamlit alongside Mediapipe for real-time pose detection. These projects often focus on building simple user interfaces where users can upload images or videos, and the pose detection results are displayed in real-time. Some implementations offer additional features such as tracking multiple people, analyzing movements, and creating interactive dashboards for sports performance analysis.

2. Fitness Apps with Streamlit

Streamlit has also been used in fitness apps to track exercises and movements in real-time. Pose detection is utilized to analyze posture during exercises like squats, lunges, and push-ups, providing feedback for improvement. Such apps are beneficial for both beginners and professionals, and the combination of Mediapipe's accuracy and Streamlit's simplicity makes them highly accessible.

2.4 Gaps in Existing Solutions

Despite the advancements in pose estimation, there are several limitations in existing solutions:

Complexity:

Tools like **OpenPose,** while accurate, require significant computational resources and setup, which limits their accessibility.

Mobile Optimization:

Many advanced systems are not optimized for real-time processing on mobile devices or low-powered hardware.

Limited Application Scope:

Existing systems often focus on specific domains, like fitness or human-computer interaction, leaving a gap for general-purpose pose detection tools that are easy to deploy and use.

User Experience:

Many tools lack intuitive, user-friendly interfaces for non-technical users. Streamlit's simplicity provides an opportunity to bridge this gap by creating easy-to-use pose detection tools.

2.5 Conclusion of Literature Survey

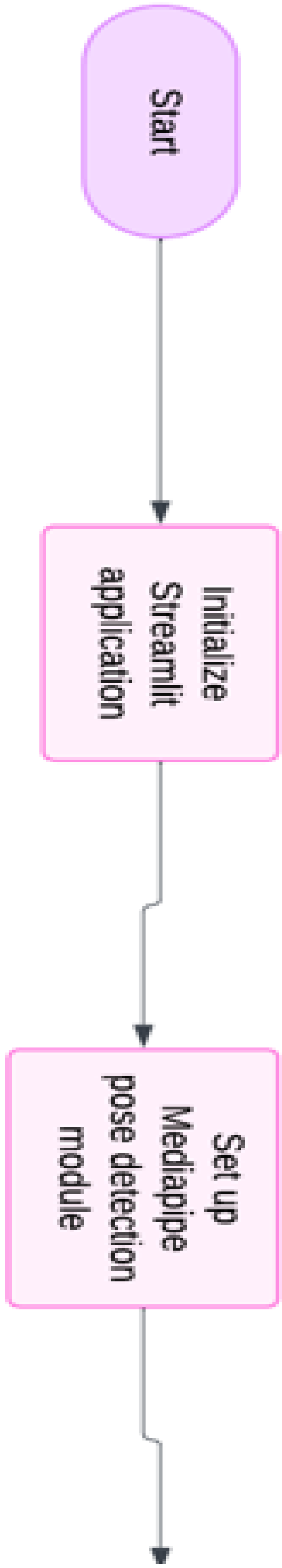
This survey reveals that while several advanced tools exist for pose estimation, most require heavy computational resources and technical expertise. The use of Mediapipe for pose estimation, coupled with Streamlit's ease of use, provides an opportunity to create a lightweight, interactive tool for pose detection that is accessible to beginners, researchers, and professionals. The integration of these technologies addresses gaps in existing solutions by offering an efficient, real-time, and user-friendly system fodetection, applicable in various domains likesports analysis, fitness, and rehabilitation.

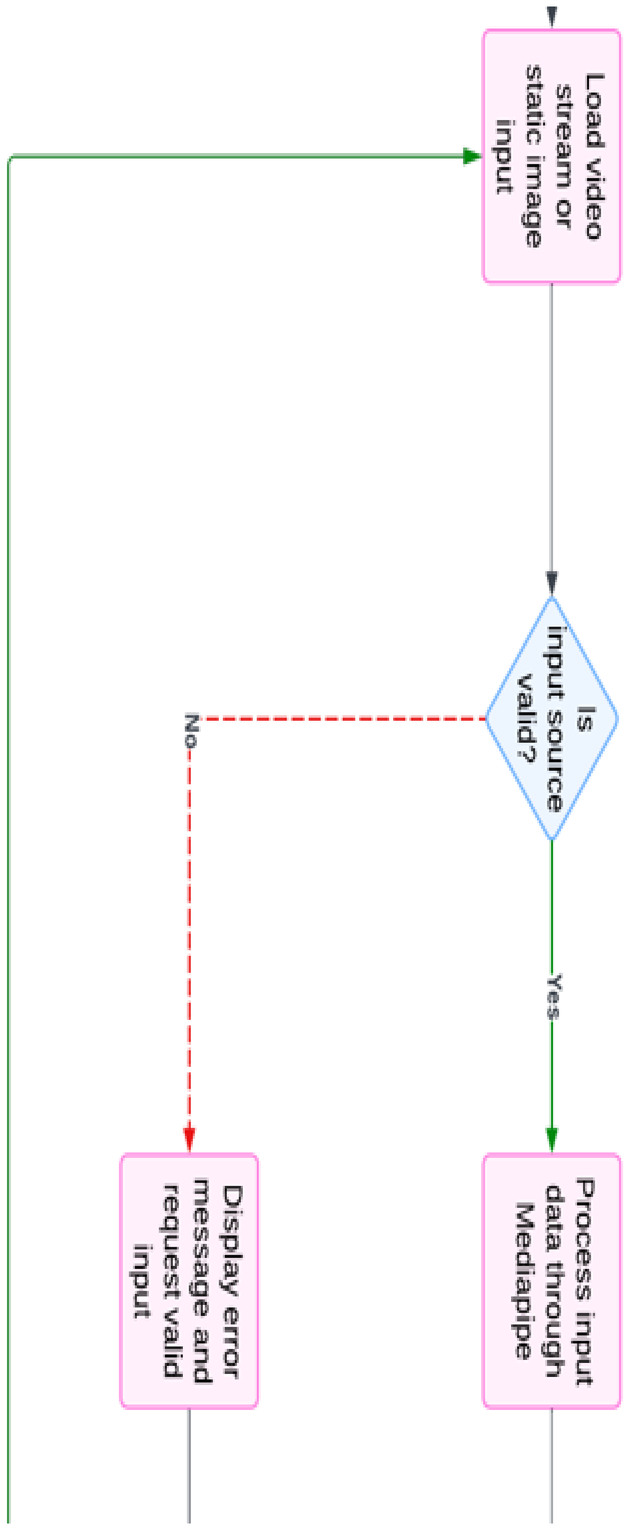
CHAPTER 3

3.1Proposed Methodology

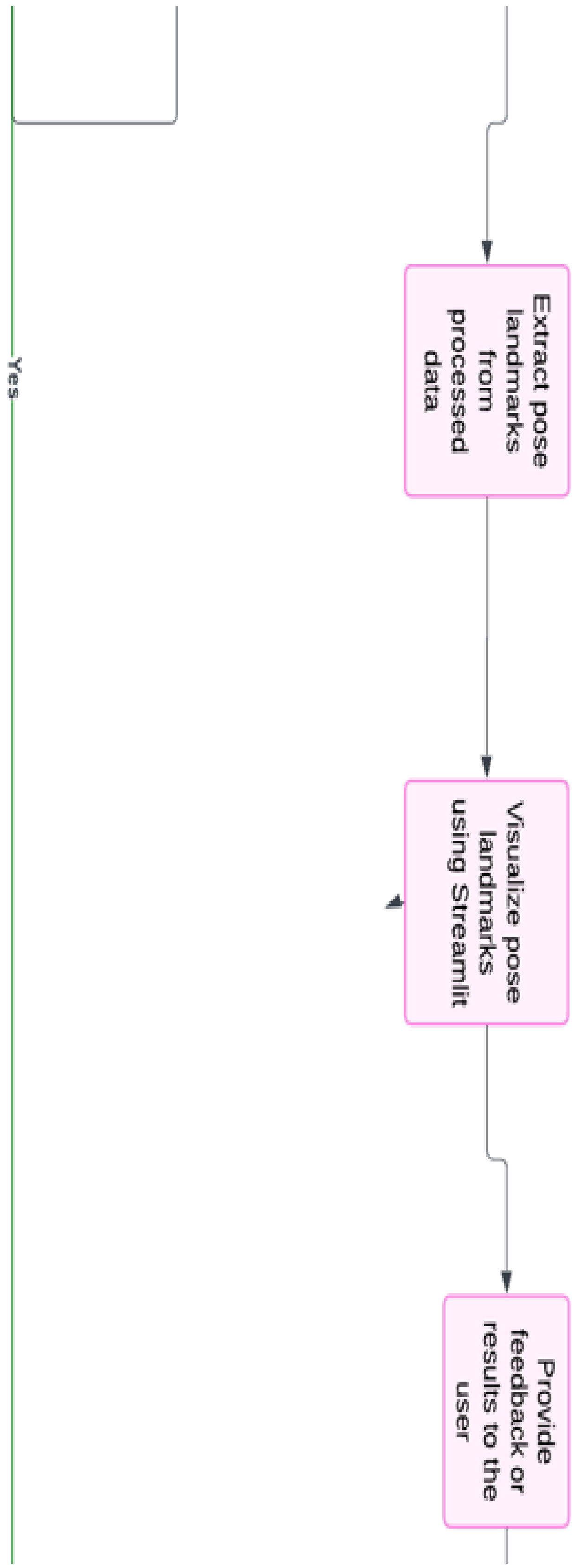
The methodology for the "Pose Detection Using Mediapipe and Streamlit" project involves several stages to develop the real-time pose detection tool. The approach is divided into the following key steps:

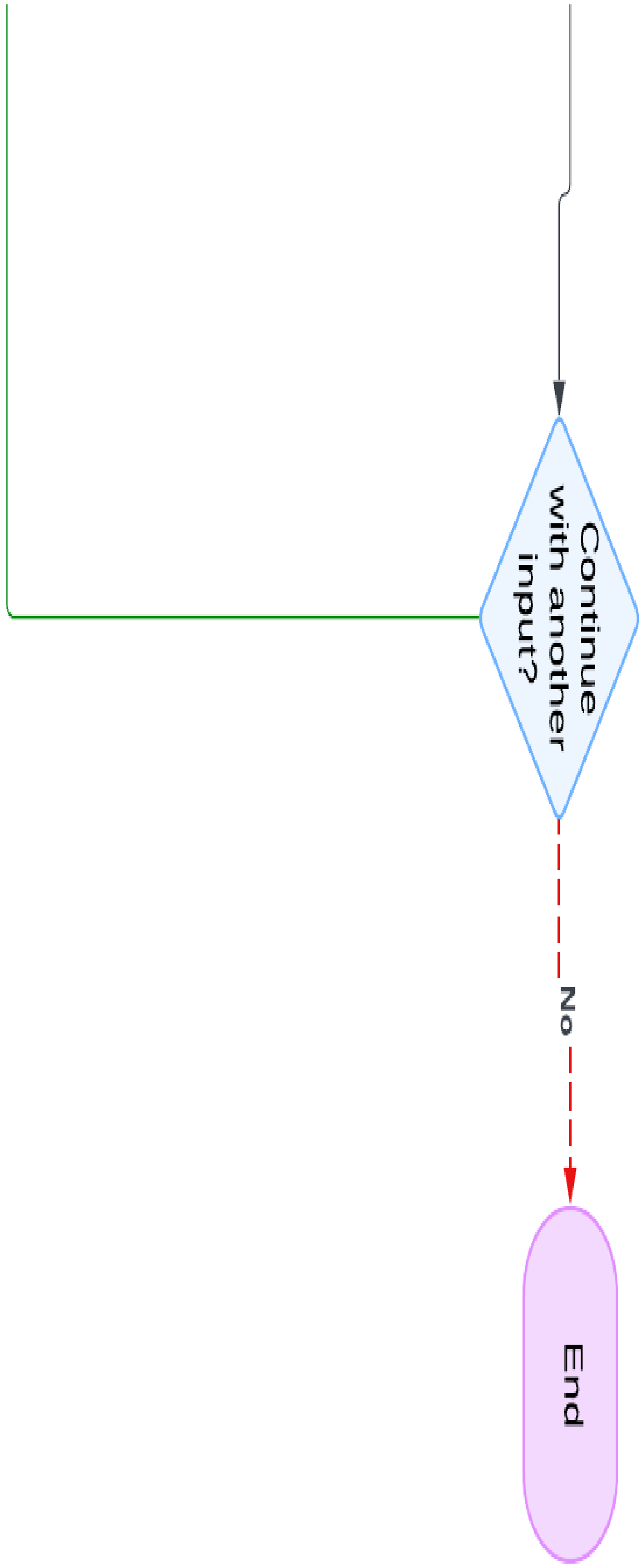
1 system design


Start
Initialize Streamlit application
Set up Mediapipe pose detection module


Load video stream or static image input
Is input source valid?
Yes
No
Process input data through Mediapipe
Display error message and request valid input

16


Yes
Extract pose landmarks from processed data
Visualize pose landmarks using Streamlit
Provide feedback or results to the user


Continue with another input?
No
End

Here’s an explanation of the flowchart described:

1. Start

- **Description**:

This is the starting point of the application. It begins when the user initiates the process in the Streamlit app.

2. Initialize Streamlit Application

- **Description:**

The Streamlit app is initialized. This step sets up the environment and creates the web interface for user interaction.

- **Action:**

This includes importing necessary libraries (e.g., streamlit, mediapipe, opencv), setting up the user interface, and creating placeholders for input and output (e.g., video stream or image upload).

3. Setup Mediapipe Pose Detection Module

- Description:

Mediapipe's pose detection module is set up here. This involves initializing the Mediapipe Pose model and specifying the parameters like min_detection_confidence and min_tracking_confidence.

- Action:

The mediapipe library is loaded, and the Pose model is initialized for processing input frames (video or image).

4. Load Video Stream or Static Image Input

- Description:

The user is prompted to upload either a video stream or a static image for pose detection.

 - Action:

The Streamlit interface allows the user to upload a file (video or image).

5. Is Input Source Valid?

 - Description:

This decision point checks whether the input provided by the user is a valid video stream or image.

- If the input is valid (i.e., the file type is supported), the program proceeds to the next step.

 - If the input is invalid (e.g., unsupported file type or empty input), the program will display an error message and prompt the user to provide valid input again.

- Action:

The program checks the file type (e.g., .mp4 for video or .jpg for image). If the input source is invalid, display an error message like "Invalid input, please upload a valid video or image."

6. Process Input Data Through Mediapipe

- Description:

If the input source is valid, the video or image is processed frame by frame (for video) or as a single frame (for an image) using the Mediapipe Pose Detection module.

- Action:

The input is converted to RGB and passed through the Pose model to detect human body landmarks.

7. Extract Landmark from Processed Data

- Description:

After processing the input data, Mediapipe detects the human body landmarks (keypoints such as joints, arms, legs, etc.).

- Action:

The landmark data (coordinates of the keypoints) is extracted and prepared for visualization.

8. Visualize Landmark Using Streamlit

- Description:

The detected landmarks are visualized in the Streamlit app.

- Action:

The landmarks are drawn on the input (image or video) using OpenCV. These landmarks are then displayed on the Streamlit interface so that the user can see them in real-time.

9. Provide Feedback

- Description:

After the landmarks are visualized, the app provides feedback to the user. This could include information such as "Pose Detected" or an error message if no pose is detected.

- Action:

Based on the result of the pose detection, feedback is given (e.g., "Pose successfully detected" or "No pose detected").

10. Continue with Another Input?

- Description:

This decision point checks if the user wants to continue with another input.

- If the user wants to continue, the program will go back to step 4 to allow uploading of a new video/image.

- If the user wants to stop, the program proceeds to the next step.

- Action:

After providing feedback, the user is prompted whether they want to analyze another video or image.

11. No (End)

- **Description:**

If the user opts to end the process, the application stops.

- **Action:**

The program terminates, and the Streamlit app is closed.

Flowchart Summary:

1. Start

2. Initialize Streamlit App→ Set up the Streamlit environment for user interaction.

3. Setup Mediapipe Pose Detection Module → Initialize Mediapipe Pose.

4. Load Video/Static Image Input→ User uploads an image or video.

5. Is Input Source Valid?→ Validate the input type (yes/no).

- If No → Display error message and request valid input.

- If Yes → Proceed to next step.

6. Process Input Data through Mediapipe → Process the video/image to extract pose landmarks.

7. Extract Landmark from Processed Data→ Extract coordinates for body landmarks.

8. Visualize Landmark using Streamlit → Display detected landmarks.

9. Provide Feedback→ Inform the user about the detection.

10. Continue with Another Input? → User chooses whether to continue or end.

  - If Yes→ Repeat from step 4.

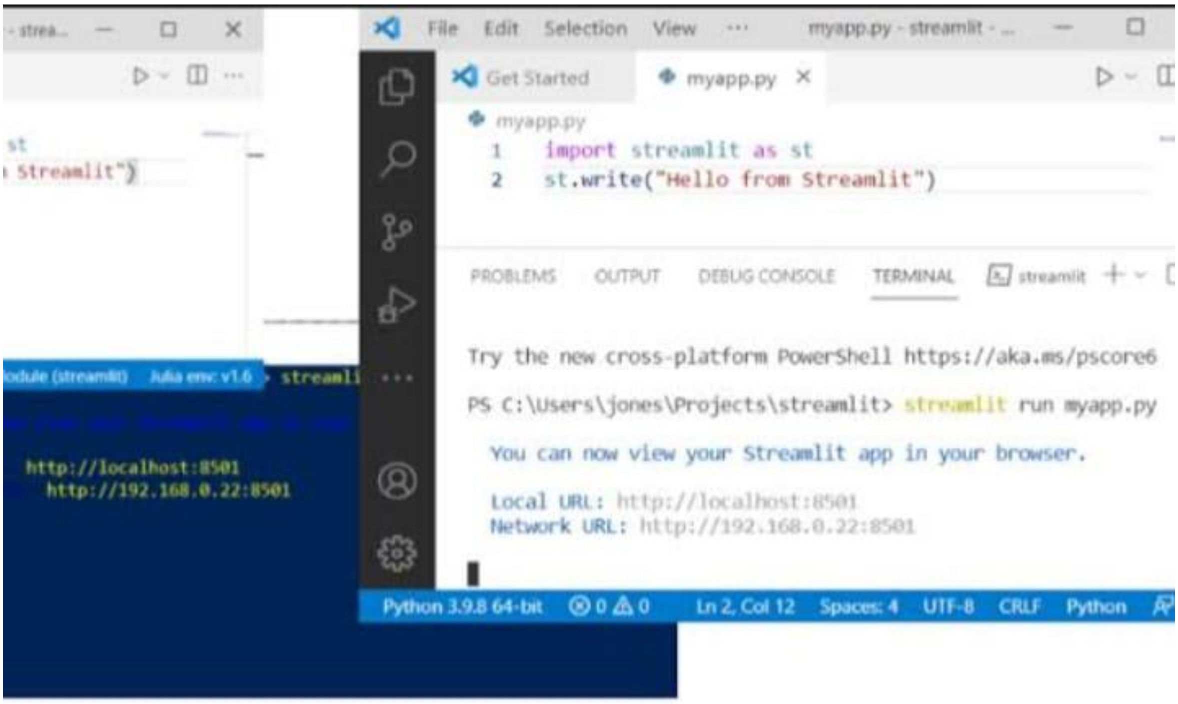
- If No→ Proceed to step 11.

11. *End* → Terminate the app.

This flowchart is a structured outline for the app's process, ensuring that all steps are handled smoothly and logically.

CHAPTER 4
Implementation and Results

Terminal command to run streamlit



STREAMLIT WEB PAGE

In the deployed Streamlit application, the *Pose Detection* model is accessible through a user-friendly web interface. The interface consists of three main sections:

1. Title Section:

At the top of the page, the title "Pose Detection" is prominently displayed, clearly indicating the purpose of the application.

2. **File Upload Section:**

Below the title, a file upload feature allows users to upload a video or an image. Users can simply drag and drop a file or browse to select one from their system. This input serves as the data source for pose detection.

3. Output Visualization Section:

After uploading, the application processes the input using Mediapipe's pose detection module. The output displays the video or image with pose landmarks highlighted (markers on joints, eyes, nose, etc.), and the joints are connected using lines for better visualization. Users can observe real-time feedback or view processed results directly in the application.

This simple yet interactive layout ensures ease of use while showcasing the results effectively.

1. Lying Touch Exercise

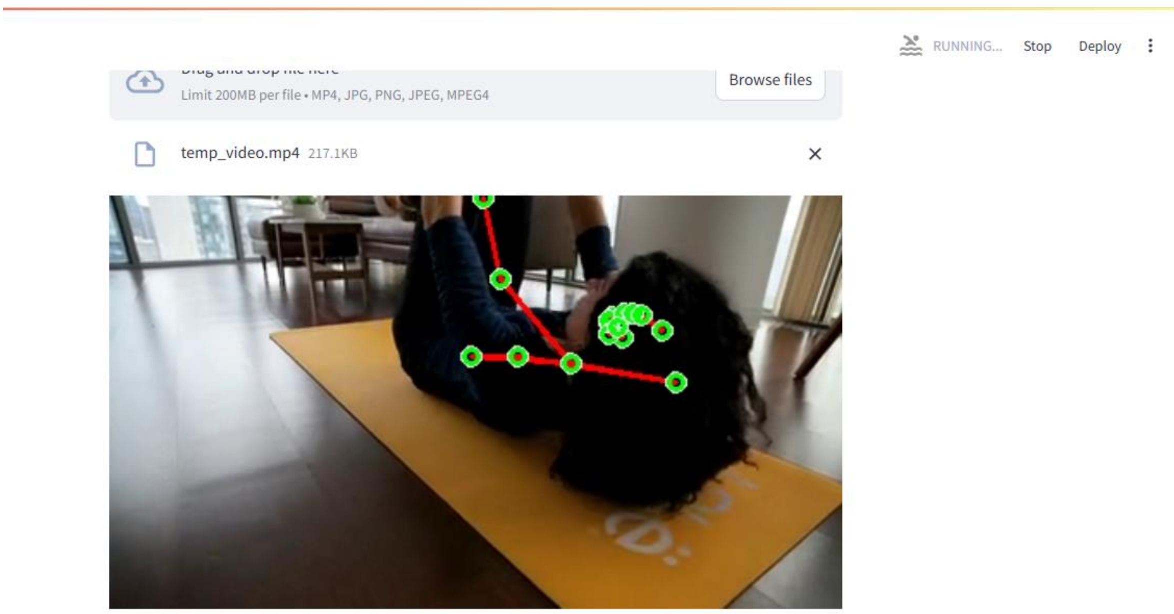
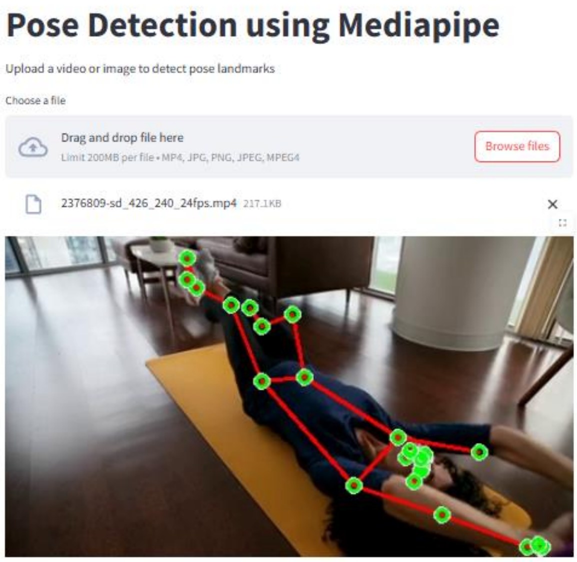


FIG 2 JOINTS ARE MARKED BY MARKERS



1. Lying Touch Exercise

-

Description of the Input:

- The individual is lying down and performing a touch-based exercise, such as raising a leg or arm to meet the opposite limb.

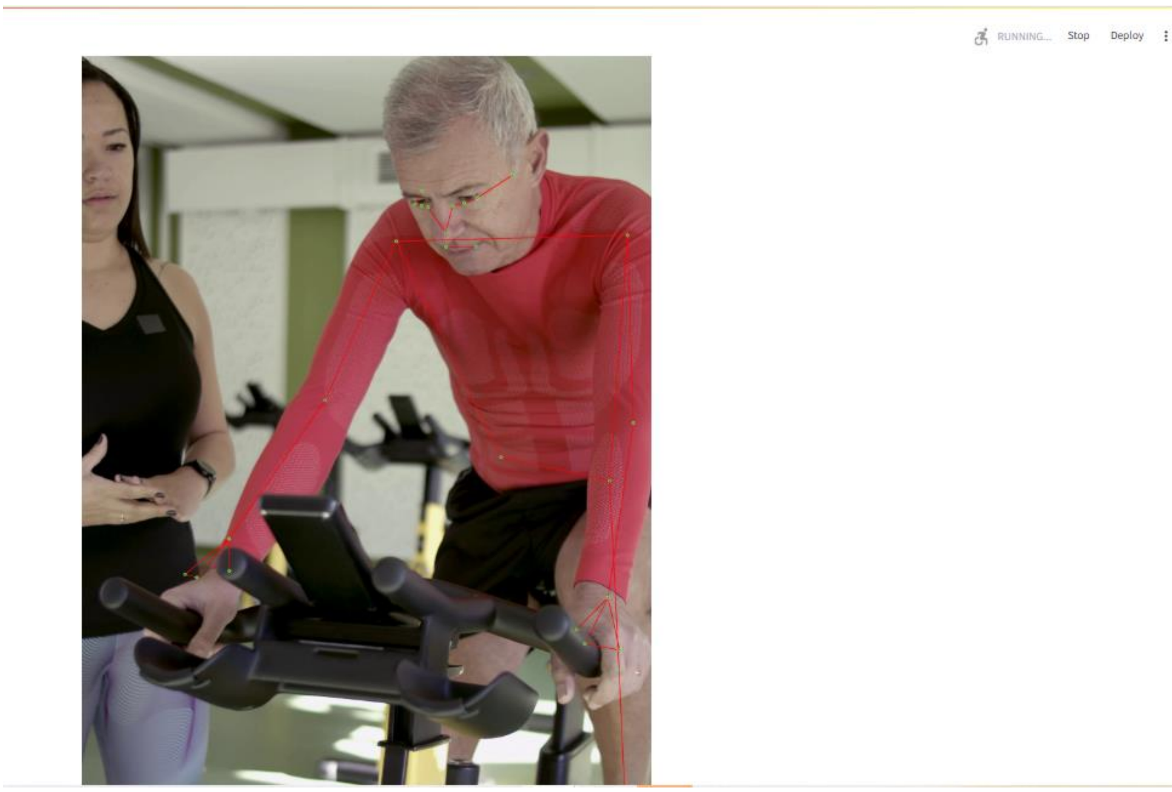
- *Pose Detection Output*:

  - Key joints like wrists, elbows, shoulders, hips, knees, and ankles are tracked.

  - The markers on the wrists and ankles highlight their movement, and lines connecting the limbs indicate the stretch or touch motion.

- *Significance*:

2. Ortho Cycle Exercise



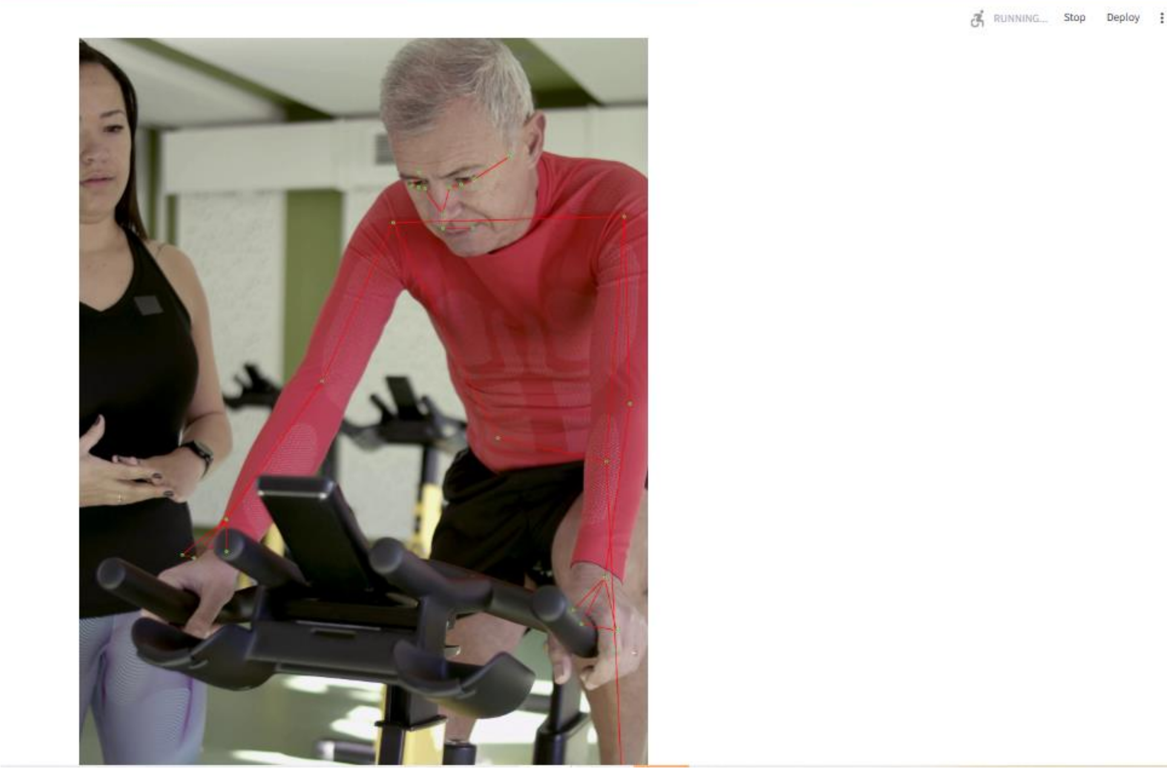


FIG 5 JONTS ARE MARKED BY SMALL GREEN MARKERS JOINEDB Y

Description of the Input:

- The input is a video of an individual performing the orthopedic cycling exercise. The individual rotates their legs as if pedaling on a bicycle.

- Pose Detection Output:

- The Mediapipe Pose Detection model detects key landmarks on the person, including knees, hips, ankles, shoulders, and elbows.

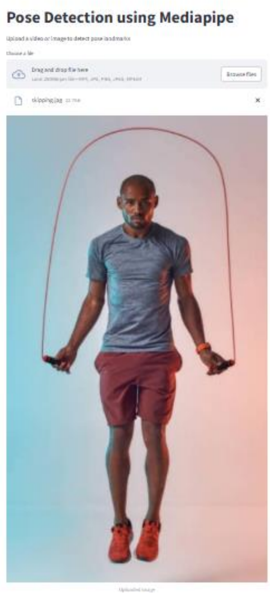
- Joints such as the knees and ankles are marked with circular markers, and these are connected by lines to represent limb movement.

- **Significance:**

- The connected lines between joints effectively illustrate the leg rotation motion. This provides visual feedback for the alignment of the knees and ankles during the exercise.

- Eye, nose, and mouth markers ensure proper body posture is maintained during the exercise.

3. Skipping Exercise*



Description of the Input:

- The video shows an individual mimicking a skipping rope motion by jumping in place.

- **Pose Detection Output:**

- - The system identifies and highlights key joints such as shoulders, elbows, wrists, hips, knees, and ankles.
- - The arms' swinging motion is tracked by the lines connecting the shoulders, elbows, and wrists, demonstrating the skipping action.

- **Significance:**

- This feedback is valuable for analyzing arm and leg coordination during skipping.

- Markers on the face (nose, mouth, and eyes) ensure that the individual maintains proper head posture while skipping.

General Analysis

- - The markers on the eyes, nose, and mouth ensure the head's position is captured, which can prevent unnecessary strain during exercises.
- - The lines between the joints provide a clear visualization of body posture and movement patterns, aiding in form correction.
- - These outputs can be extended for applications in fitness monitoring, physical therapy, or even sports training, allowing users to achieve better results while minimizing injuries.

You can use this explanation as a detailed guide for each exercise's output. Let me know if you need help adding more specific points!

4.1 GitHub Link for Code:

https://github.com/ARCYBORG-005/HUMAN-POSE-ESTIMATION-ML.git

CHAPTER 5

Discussion and Conclusion

5.1 Future Work:

- *Improving Model Accuracy:* Enhance the pose detection model by fine-tuning Mediapipe's parameters or integrating advanced algorithms to handle diverse body movements and poses more accurately.

- *Real-time Performance Optimization:* Optimize the application for faster processing to support real-time pose detection, making it suitable for live sports analysis or fitness tracking.

- *Support for Additional Features:* Extend the application to include activity recognition, posture correction suggestions, or automatic feedback based on detected poses for fitness or rehabilitation purposes.

- *Compatibility with More Input Formats:* Improve compatibility by supporting various video codecs and image formats, ensuring smooth user experience across platforms.

- *Mobile and Cloud Deployment:* Deploy the application on mobile platforms or cloud services, making it more accessible and scalable for diverse users.

5.2 Conclusion:

- This project demonstrates the successful implementation of a pose detection system using Streamlit and OpenCV, powered by Mediapipe's robust pose estimation technology.

- The application effectively identifies and visualizes human body landmarks, providing accurate feedback on body posture through marker-based visualization.

- The project's interactive interface enables users to analyze video or image inputs seamlessly, making it a valuable tool for applications in sports, fitness, and rehabilitation.

- The work contributes significantly to the field by simplifying pose detection workflows while ensuring ease of deployment and user interaction.

- Future enhancements can further expand its functionality, making it even more impactful in real-world scenarios.

REFERENCES

1.Ming-Hsuan Yang, David J. Kriegman, Narendra Ahuja, “Detecting Faces in Images: A Survey”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume. 24, No. 1, 2002.

1. *Mediapipe Library:*
    a. Zhang, F., Bazarevsky, V., Vakunov, A., Sung, G., Grundmann, M., & Radosavovic, I. (2020). Mediapipe: A cross-platform framework for building multimodal applied ML pipelines. https://google.github.io/mediapipe/
2. *OpenCV Library:*
    a. Bradski, G., & Kaehler, A. (2000). OpenCV: Open Source Computer Vision Library. https://opencv.org/
3. *Streamlit Framework:*
    a. Streamlit Inc. (2019). Streamlit: The fastest way to build and share data apps. https://streamlit.io/
4. *Python Programming Language:*
    a. Python Software Foundation. (2025). Python Language Reference, Version 3.10. https://www.python.org/
5. *Matplotlib Library:*
    a. Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95.
6. *Project-Specific Dataset:*
    a. The video and images used for pose detection analysis were sourced from publicly available datasets and user-generated inputs to test functionality.