GymGuardian - Fitness form corrector and Rep counter using OpenCV and Mediapipe

A PROJECT REPORT

Submitted by

Yash Mehta (24BCE10638)

Kartikey Karanwal (24BCE10508)

Harsh Ratnaparkhe (24BCE10892)

Rudradeep Chakraborty (24BCE11151)

Prathamesh Kapil Dhakad (24BCE10301)

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

VIT BHOPAL UNIVERSITY KOTHRI KALAN, SEHORE

MADHYA PRADESH - 466114

September 2025

VIT BHOPAL UNIVERSITY, KOTHRI KALAN, SEHORE MADHYA PRADESH – 466114

BONAFIDE CERTIFICATE

Certified that this project report titled **"GymGuardian - Fitness form corrector and Rep counter using OpenCV and Mediapipe"** is the bonafide work of "**Harsh Ratnaparkhe (24BCE10892), Rudradeep Chakraborty (24BCE11151), Prathamesh Kapil Dhakad (24BCE10301), Yash Mehta (24BCE10638), Kartikey Karanwal (24BCE10508)"** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported at this time does not form part of any other project/research work based on which a degree or award was conferred on an earlier occasion on this or any other candidate.

PROGRAM CHAIR
Dr. Vikas Panthi
School of Computer Science and Engineering
VIT BHOPAL UNIVERSITY

PROJECT GUIDE
Dr. Dheresh Soni, Program Chair
School of Computer Science and Engineering
VIT BHOPAL UNIVERSITY

The Project Exhibition I Examination is held on _____

ACKNOWLEDGEMENT

First and foremost I would like to thank the Lord Almighty for His presence and immense blessings throughout the project work.

I wish to express my heartfelt gratitude to **Dr. Vikas Panthi**, Program Chair, SCOPE for much of his valuable support and encouragement in carrying out this work.

I would like to thank my internal guide **Dr. Dheresh Soni**, Program Chair, SCOPE for continually guiding and actively participating in my project, giving valuable suggestions to complete the project work.

I would like to thank all the technical and teaching staff of the School of Computer Science and Engineering, who extended directly or indirectly all support.

Last, but not least, I am deeply indebted to my parents who have been the greatest support while I worked day and night for the project to make it a success

LIST OF ABBREVIATIONS

Abbreviations	Full Forms
Mediapipe	Pose Landmarks Detection Library
Open CV	Open Source Computer Vision Library
lmList	Landmark List (list of detected body landmark coordinates)
FPS	Frames Per Second
UI	User Interface
AI	Artificial Intelligence
ML	Machine Learning

LIST OF FIGURES

LIST OF GRAPHS

LIST OF TABLES

ABSTRACT

The growth of at-home fitness has highlighted a significant gap in the safety and effectiveness of workouts and that is a lack of real-time corrective feedback. This project "Gym Guardian" provides a unique solution develop an interactive virtual personal trainer that uses computer vision to analyse and provide feedback on a user's exercise form in real-time. The system captures user video with an existing cameral and processes it through OpenCV for frame handling and the Mediapipe framework for high-fidelity detection of pose landmarks. By identifying the angles of key body joints through vector math, the system can measure a user's posture against biomechanically safe ranges for a specific exercise. A user will receive immediate and actionable feedback displayed through a graphical user interface, which includes visual overlays, form accuracy feedback, and an automated repetition count feature. In aggregate, this project will provide a democratized approach to personal training by providing an affordable, accessible, and accurate option that allows someone to exercise injury-free and efficiently, while maximizing the benefits of the workout.

TABLE OF CONTENTS

CHAPTER 1: PROJECT DESCRIPTION AND OUTLINE

1.1 Introduction

Gym Guardian is a computer vision-based software that acts as a virtual personal trainer. The application analyses a user's exercise form in real-time through a webcam and can provide instant corrective feedback. Specifically, machine learning models for pose estimation are utilized in Gym Guardian to identify a user's body joints, dimensions, and angles, which are then compared to ideal angles for different exercises. This provides users, especially novice users, the ability to complete workouts correctly and ensure effective movements that significantly limit the risk of injury associated with poor movement patterns.

1.2 Motivation For Work

There are two key challenges currently facing the fitness space motivating this project. First, expanding numbers of people are working out at home without any supervision. This is a concern as inexperienced exercisers are much more likely to hurt themselves due to using improper form. Second, certified personal trainers do a great job of helping people exercise safely and effectively, but the financial costs and scheduling challenges associated with using them limit people's access to their services. Thus, there exists a niche for providing an automated, low-cost, and easily accessible solution that can provide guidance from an expert and to fill the gap Gym Guardian aims to assist all individuals in continuing to exercise with safety and effectiveness.

1.3 About Introduction to the Project (Including Techniques)

The GymGuardian AI personal trainer is built on the foundation of computer vision and real-time human pose estimation techniques. It uses a standard webcam to capture the user's full body movements in real-time. Core techniques, such as MediaPipe's advanced pose estimation model and OpenCV-based image processing, enable the accurate detection and tracking of key body landmarks like shoulders, elbows, and knees. The system leverages custom algorithms to analyse these landmarks, mapping body movements to biomechanical data such as joint angles and relative positions. This data is then used to provide automated rep counting and real-time form correction. This integration of an advanced AI model with exercise-specific logic creates a seamless and robust feedback loop, effectively turning a simple webcam into a smart virtual coach.

1.4 Problem Statement

Many individuals exercising at home, especially beginners, lack access to proper feedback that too in real time for their form, leading to a high injury and diminished results. The high cost

and scheduling constraints of personal trainers make them inaccessible to a large audience, creating a need for an affordable and accurate solution for immediate, corrective guidance.

1.5 Objective of the Work

Our objective is to create a virtual trainer that provides feedback on exercise form. By offering immediate technique correction, we empower individuals to work out safely and effectively at home, eliminating the guesswork and high risk of injury associated with unguided exercise.

1.6 Organization of the Project

The report is structured into seven chapters, each covering a key aspect of the project:

1. **Chapter 1**: Provides overview of the project, objectives and organizational structure.
2. **Chapter 2**: Investigates related work in the field, identifying existing methods.
3. **Chapter 3**: Discusses the technical and functional requirements for the system.
4. **Chapter 4**: Describes the design methodology and the system's novel features.
5. **Chapter 5**: Outlines technical implementation and analyses the system's performance.
6. **Chapter 6**: Highlights the project's outcomes, applicability, and significance.
7. **Chapter 7**: Concludes the project with recommendations for future enhancements.

1.7 Summary

This chapter introduced the Gym Guardian project, a virtual fitness coach designed to provide real-time form correction. It outlined the motivation behind the project, stemming from the need for safe and accessible home fitness guidance. The core technologies—OpenCV for video processing, Mediapipe for pose estimation, and NumPy for angle calculation—were introduced. The chapter concluded with the formal problem statement and objectives that guide the project's development.

CHAPTER 2: RELATED WORK INVESTIGATION

2.1 Introduction

It's clear that technology has become a huge part of our fitness journeys. We're all looking for a personal trainer experience: getting guidance to make sure we're exercising safely and effectively while also spending as little as possible. The traditional solution has always been the human personal trainer. While incredibly effective, hiring a personal trainer can be a luxury due to high costs and scheduling challenges, making it inaccessible for a large portion of the population.

This project steps into that gap, exploring how modern computer vision can serve as an accessible and intuitive virtual fitness coach. By using technology already available in most homes, a simple webcam, We can offer real time, data-driven feedback on physical exercises, making personalized fitness guidance more affordable and convenient for everyone.

2.2 Core Area Of The Project

Our central goal is to bring real-time exercise analysis to everyone. The system uses a standard webcam to run a dedicated pose estimation model, which identifies and tracks the body's key landmarks (shoulders, elbows, hips, etc.). This isn't passive recording, it's active analysis. By calculating the geometric relationship between these landmarks—specifically the joint angles—the system provides immediate, quantitative feedback on form and accurately counts repetitions, effectively converting a simple camera into a data-driven virtual coach.

2.3 Our Approach: A Virtual Fitness Coach

Our system works by turning a simple webcam into a smart observer that can understand and guide human movement. Here’s a breakdown of how it achieves this:

- **Creating a Digital Skeleton (Pose Estimation)**
  - Instead of looking for colors or simple shapes, our system takes help of the MediaPipe library for pose estimation and landmarks detection.
  - It analyses the video feed in real-time to identify and track the user's key body landmarks (like shoulders, elbows, and knees), essentially creating a live, digital skeleton of the person exercising.

- **Understanding the Movement (Biomechanical Analysis)**
  - With the digital skeleton in place, the system can measure the angles of the joints. For a bicep curl, for an instance, it calculates the angle of the elbow using the positions of points at the shoulder, elbow, and wrist landmarks.

    - This allows the system to understand the user's range of motion in brief, whether their arm is fully extended or fully flexed.

- **Counting Reps Intelligently**
    - The system translates the measured joint angle into a simple 0-100% progress scale using a mathematical technique called linear interpolation.
    - A repetition is counted in two phases. It adds half a rep when the user reaches the peak of the exercise (e.g., >95% curl) , and the other half when they return to the starting position (e.g., <5% curl). This two-step process ensures a full and proper range of motion is completed for each rep counted.

- **Providing Smooth, Real-Time Feedback**
    - The system provides immediate visual feedback through an on-screen interface, including a dynamic progress bar and a clear rep counter.
    - To make the progress bar's movement look natural and less jerky, a smoothing factor is applied. This makes the visual feedback fluid and easier to follow as the user performs the exercise.

2.3.1 Method 1: Automatic Body Tracking

Our system is designed to be incredibly simple. It uses your webcam to automatically find and track key points on your body- like your shoulders, elbows, and knees. These points, called landmarks, are then organized into a list (lmList) that contains their exact on-screen coordinates. This all happens instantly, so you can just start your workout without any complicated setup.

2.3.2 Method 2: Powered by a Pre-Trained AI

The magic behind this automatic tracking is a powerful, pre-trained AI model from Google's MediaPipe library. Instead of building an AI from scratch, we use this ready-made, highly efficient model that is an expert at one thing: human pose estimation. This allows the entire system to run smoothly in real-time, even on a regular home computer.

2.3.3 Method 3: Understanding Movement with Geometry

Once the system knows where your joints are, it begins to understand *how* you're moving. It does this by using basic geometry to calculate the angles of your joints. For a squat, it constantly measures the angle of your knee to see how deep you're going. This real-time angle measurement is the key piece of data that tells the system if you've completed a full, proper repetition.

2.4 Pros And Cons Of The Stated Methods

PROS	CONS
Highly Accessible: Anyone with a standard webcam can use it, making fitness guidance affordable and removing the need for expensive equipment or sensors.	**Environment Dependent:** The program's accuracy can be affected by poor lighting or a very cluttered background that confuses the model.
Easy to Use: The system works automatically without needing any physical markers or complicated setup. You can just launch the application and start your workout.	**2D Limitations:** A normal webcam sees in 2D, so it can't perfectly judge depth. This makes it difficult to detect subtle form issues like spinal rotation or if knees are caving inward.
Objective Feedback: It provides real, data-driven numbers—like joint angles and rep counts—which helps in tracking progress more accurately than just looking in a mirror.	**Landmark Occlusion:** If one body part blocks the camera's view of another, the model can temporarily lose track, potentially leading to inaccurate measurements.
Instantaneous Guidance: Users get immediate visual feedback, allowing them to correct their form as they exercise, which is crucial for maximizing effectiveness and preventing injury.	**Performance Varies:** The system's smoothness (FPS) depends on the user's computer. Older or less powerful machines might experience some lag in the video feedback.

Table 2.1 : Pros and cons of stated methods

2.5 Issues/Observation From Investigation

Based on our development, we've noted a few key things about this approach:

- **Camera Angle is Key:** The system's accuracy is highly dependent on the user's position relative to the camera. For most exercises, a direct front-facing view works best. For others, like push-ups, a clear side profile is necessary to get meaningful angle measurements. An awkward camera angle can skew the results.
- **The "Jitter" Effect:** During very fast movements or if the user wears baggy clothing, the AI model can sometimes have trouble locking onto the exact landmark location, causing the on-screen skeleton to "jitter." We implemented a smoothing factor in the UI to counteract this and make the feedback feel more stable and fluid.
- **Simplifying "Good Form":** One of the biggest challenges is boiling down the complex, nuanced idea of "perfect form" into simple geometric rules (i.e., angle

thresholds). While effective for counting reps, these rules are a simplification and can't capture every detail a human coach would.

- **Balancing Quality and Speed:** We found a constant trade-off between video quality and real-time performance. While a higher resolution image can lead to better landmark detection, it demands more processing power. We settled on a 720p resolution as the sweet spot to ensure the application remains responsive and smooth for most users.

2.6 Summary

In this chapter , we have done a deep dive into where our project lies in the field of fitness enhanced with technology, where we explore a cost effective yet reliable alternative to a traditional personal trainer. Mediapipe played a crucial role in identifying the landmark points which are then calculated against and then subsequently the joint angles are too calculated. The data then translates to repetitions and provides real time feedback to the user directed towards their exercise form. Lastly , this chapter looked at the pros and cons of the visual approach highlighting the physical limitations such as lighting dependence, camera placement.

CHAPTER 3: REQUIREMENT ARTIFACTS

3.1 Introduction

In this section we shall take a look into the various technical requirements in terms of both software as well as hardware ; Diving deep into the various technicalities and limitations thereof for the implementation of “Gym Guardian”.

3.2 Hardware and Software Requirements

This section outlines the specific hardware, software, functional, and performance requirements necessary for the successful implementation and operation of the Gym Guardian system. These requirements serve as the technical blueprint for development and the benchmark for evaluation.

Hardware Requirements:

To support the pose landmark detection and reps counter, the following hardware components are necessary:

1. Camera:

- A high-resolution RGB camera (preferably at least 720p or 1080p) for capturing clear images of the body and its various landmarks.
- A camera with high frame rates (at least 30 fps) is recommended to enable real-time tracking and landmark detection consistently.

2. Computer/Processing Unit:

- A computer with sufficient processing power to handle image processing tasks in real-time.

Minimum specifications:

- Processor: Intel i5 or equivalent (quad-core or higher).
- RAM: At least 8GB of RAM.
- Graphics: A discrete GPU (e.g., NVIDIA GTX series) can enhance performance but is not mandatory.
- Storage: SSD with at least 1-2 GB of available space for storing program files and images.

3. Lighting Setup:

- Good lighting is crucial for consistent landmark detection. It can ensure the accuracy to be high and make the rep counter perform better.

Software Requirements:

1. **Operating System:**
    - Windows 10 or higher, or macOS (for compatibility with computer vision libraries).
2. **Programming Languages:**
    - Python: Primarily used for image processing and computer vision tasks.
3. **Libraries/Frameworks:**

    Core Libraries:

    - **OpenCV (cv2):** This library is essential for all real-time video processing tasks. In this project, it is used to capture the video feed from the webcam, perform image manipulations like resizing and flipping, and render all the visual feedback elements (e.g., progress bars, text, and landmark connections) onto the video frames for the user to see.
    - **NumPy:** A fundamental library for numerical computation. It is specifically used for its efficient array operations and mathematical functions, most notably the linear interpolation function (np.interp), which maps the calculated joint angles to user-friendly percentage values and dynamic progress bar heights.
    - **MediaPipe:** This is the core framework that provides the advanced machine learning models for pose estimation. It accurately detects and tracks the user's key body landmarks from the video feed in real-time, forming the foundation for all exercise analysis and rep counting.

    GUI and Deployment:

    - **Tkinter:** This is Python's standard library for creating graphical user interfaces (GUIs). It will be implemented to build a user-friendly main menu, allowing users to easily select which exercise they want to perform without running different scripts manually.
    - **PyInstaller:** This tool will be used to package the entire GymGuardian application—including the Python scripts and all the required libraries—into a single, standalone executable (.exe) file. This makes the application portable and easy for end-users to run on Windows without needing to install Python or any dependencies.
4. **IDE:**
    - A suitable integrated development environment (IDE) for coding and debugging, such as VSCode.
5. **Version Control:**

- Git: For managing code versions and collaboration.

3.3 Specific Project Requirements:

3.3.1 Data Requirements

- **Video Frame Data:** The system requires a real-time video stream from a standard webcam (at least 720p resolution) to capture the user's movements clearly. The quality of this data is crucial for the accuracy of landmark detection.
- **Pose Landmark Data:** The system processes video frames to generate 2D coordinate data for key body landmarks (e.g., shoulders, elbows, hips, knees) using the MediaPipe Pose model. This stream of landmark data is the primary input for all subsequent calculations.
- **Derived Biomechanical Data:** This is data calculated in real-time from the raw landmark positions. It includes joint angles (e.g., the angle of the elbow during a bicep curl) or relative positional deltas (e.g., the vertical distance between shoulders and hips during a pull-up). This data is used to quantify the user's exercise form and progress.
- **Operational Environment Data:** For optimal performance, the system requires data from an environment with adequate, consistent lighting and a non-cluttered background. This minimizes interference and improves the model's ability to accurately detect the user.

3.3.2 Functional Requirements

- **Real-time pose estimation:** The system must capture the video feed and accurately detect the user's body landmarks in real-time, processing the stream at a sufficient frame rate (ideally 20-30 FPS) to ensure smooth tracking.
- **Exercise Repetition Counting:** The core function is to analyse the derived biomechanical data to accurately count exercise repetitions. The system identifies a full rep by tracking the user's movement through the complete range of motion—from full extension to full flexion and back.
- **Real-Time Feedback:** The system must provide immediate visual feedback to the user. This includes an on-screen repetition counter and a dynamic progress bar that visually represents the user's current position within the defined range of motion for the exercise.
- **Form and Range of Motion Analysis:** The system must define and enforce a correct range of motion for each exercise using specific angle thresholds (e.g., FULL_FLEXION_ANGLE, FULL_EXTENSION_ANGLE). For certain exercises like push-ups, it must also provide corrective feedback on posture by monitoring angles related to the neck and lower back.
- **User Feedback Interface:** The system must present all information through a clear and intuitive graphical interface overlaid on the user's video feed. This interface will display the exercise name, repetition count, progress bar, percentage of completion, and any form-specific feedback without obstructing the user's view of their own body.

3.4 Summary:

This chapter has outlined the basic specifications for the Gym Guardian system. The specifications include both the required commodity hardware, such as the standard webcam, and the specified Python libraries that constitute the software stack (OpenCV , Mediapipe , and NumPy). The functional requirements described the basic functions of the Gym Guardian system including video capture, pose landmark detection, the mathematical calculation of joint angles , the form evaluation logic, automated counting of repetitions, and real-time visual feedback .

CHAPTER 4: DESIGN METHODOLOGY AND ITS NOVELTY

4.1 Methodology and Goal

Gym Guardian's approach is based on a modular, pipeline architecture that processes video data in sequence to enable real-time analysis of user form. The main intention of this work is to create a reliable and extendable platform that can accurately guide users through exercise using only a webcam for input from the user. The novelty of this approach is not in the invention of new pose estimation algorithms and methods, but in using and adapting current state-of-the-art tools (Mediapipe) into a retail fitness application that meets a real-world market demand.

Key Methodology Components:

1. **Input Acquisition:** Continuously capture video frames from a webcam.
2. **Preprocessing:** Resize and flip frames for consistent processing and a natural mirror-like view for the user.
3. **Pose Estimation:** Apply the Mediapipe Pose model to each frame to detect body landmarks.
4. **Kinematic Analysis:** Calculate joint angles from the landmark coordinates.
5. **State Evaluation:** Use the calculated angles to determine the current state of the exercise (e.g., extension, flexion) and update the repetition count.
6. **Feedback Generation:** Translate the analysis into intuitive visual feedback displayed as an overlay on the video feed.

4.2 Functional Modules Design and Analysis

While the current implementation for each exercise is largely self-contained within a single script, for the purpose of design and analysis, the system's functionality can be broken down into the following distinct logical modules. This conceptual separation helps to clarify the flow of data and the specific responsibilities at each stage of the process, from capturing the initial video to rendering the final feedback on the screen.

1. Video Input & Pre-processing Module

- **Input:** Raw video stream from the user's webcam.
- **Process:** This module captures each frame of the video. It then resizes the frame to a standard 720p resolution for consistent performance and flips it horizontally to provide an intuitive, mirror-like view for the user.

- **Output:** A clean, standardized video frame prepared for the AI model.

2. Pose Estimation Module

- **Input:** A single pre-processed video frame.
- **Process:** The frame is fed into the MediaPipe pose estimation model. This powerful AI analyses the image to identify and output the precise on-screen coordinates of 33 key body landmarks, creating a digital map of the user's body.
- **Output:** A list (lmList) containing the coordinates of each detected body landmark.

3. Biomechanical Analysis Module

- **Input:** The list of body landmark coordinates.
- **Process:** Using the coordinates for specific joints, this module applies geometric calculations to measure the key metric for an exercise. For most workouts, this involves calculating the angle of a joint (e.g., the knee in a squat). For others, it might be the distance between two points (e.g., shoulders and hips in a pull-up).
- **Output:** A real-time numerical value (e.g., angle in degrees) that quantifies the user's movement.

4. Exercise Logic & Repetition Counting Module

- **Input:** The real-time biomechanical value (e.g., the joint angle).
- **Process:** This is the brain of the system. It first translates the input angle into a simple 0-100% range of motion score. It then uses state-tracking logic to monitor whether the user is going up or down in the movement. A half-rep is counted at the peak of the motion and the other half is counted upon returning to the start, ensuring a full, proper repetition is completed.
- **Output:** The updated repetition count and the current completion percentage.

5. User Feedback & Rendering Module

- **Input:** The original video frame and all calculated data (rep count, percentage, etc.).
- **Process:** This module uses the calculated data to draw all the visual elements for the user. It overlays the repetition counter, a dynamic progress bar, the percentage text, and other helpful information directly onto the video stream. It also applies a smoothing algorithm to the visuals to ensure they aren't jerky or distracting.
- **Output:** The final video frame with all the feedback, which is displayed to the user.

4.3 Software Architectural Design

The software architecture for GymGuardian is designed using a layered approach. This separates the system's core responsibilities into distinct logical layers, ensuring a clear and organized flow of data from video capture to user feedback.

1. Input Layer (Data Acquisition)

- This layer is the system's connection to the physical world.
- Its primary responsibility is to capture the live video feed from the user's webcam using the OpenCV library.
- It also handles initial image pre-processing, such as resizing each frame to a consistent 720p resolution and flipping it horizontally to create an intuitive mirror-view for the user.

2. Processing Layer (Core Engine)

- This is the brain of the application where all the analysis occurs. It receives prepared video frames from the Input Layer.
- Pose Landmark Detection: It uses the MediaPipe model to analyse the frame and identify the coordinates of the user's key body landmarks.
- Biomechanical Analysis: Using these coordinates, it calculates critical metrics for the exercise, such as the angle of a specific joint.
- Exercise Logic and Rep Counting: It applies a set of rules to the calculated angle to track the user's progress through a repetition, intelligently counting reps only when a full range of motion is completed.

3. Presentation Layer (Visual Feedback)

This layer is responsible for communicating all the processed information back to the user.

- **UI Rendering:** It takes the data from the Processing Layer (like the current rep count and percentage) and uses OpenCV's drawing functions to overlay this information onto the video frame.
- **Display:** It renders the final, annotated video stream in the application window, providing the user with real-time, visual guidance on their performance.

4.4 Subsystem Services

1. Real-Time Pose Tracking

- Provides a continuous, real-time stream of the user's body landmark coordinates, captured directly from the webcam feed. This foundational service enables all other analytical functions of the system.

2. Automated Repetition Counting

- Offers an automated and accurate counting service for a variety of exercises like bicep curls, squats, and pull-ups. The service intelligently identifies a completed repetition by analysing the user's movement through a full range of motion.

3. Form and Range of Motion Analysis

- Delivers real-time analysis of the user's exercise form. This includes mapping their movement to a 0-100% completion score based on predefined angle thresholds and, for certain exercises, providing specific postural feedback, such as neck and lower back alignment during push-ups.

4. Dynamic Visual Feedback

- Renders a comprehensive, real-time visual guide directly on the user's video feed. This service includes displaying the current rep count, a dynamic progress bar visualizing the range of motion, and other contextual information to keep the user informed and engaged during their workout.

5. Exercise Selection and Management

- Provides a user-friendly interface for selecting from the list of available exercises. This service will allow the user to easily start, stop, and switch between different workout routines, centralizing the management of the application.

4.5 User Interface Designs

The user interface (UI) is designed to be simple and intuitive:

The figure 4.1 shows how GymGuardian detects the landmarks on user's legs and calculate the knee angle for determining a rep for lunges.

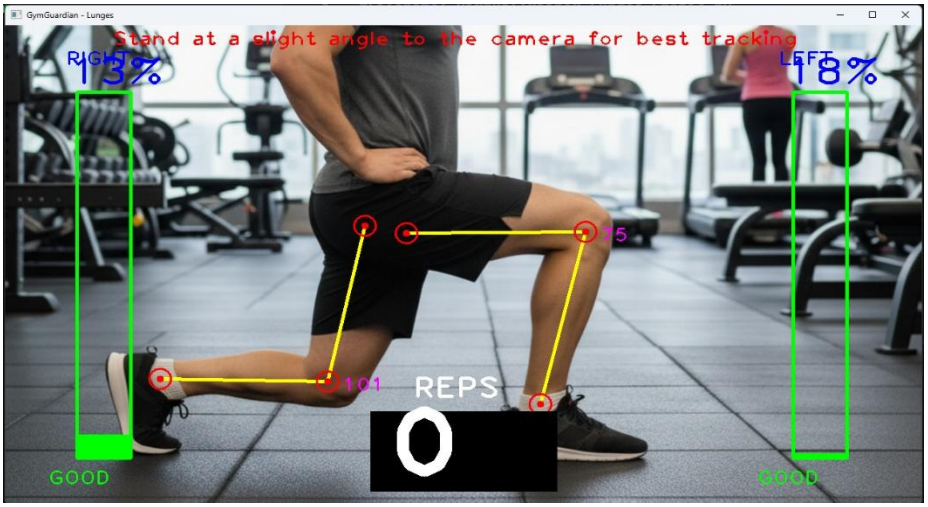


Fig. 4.1 : Lunges
(Demonstrating angle calculation and landmark detection during lunges)

The figure 4.2 shows how GymGuardian detects the landmarks on user's legs and calculate the knee angle for determining a rep for pullups.

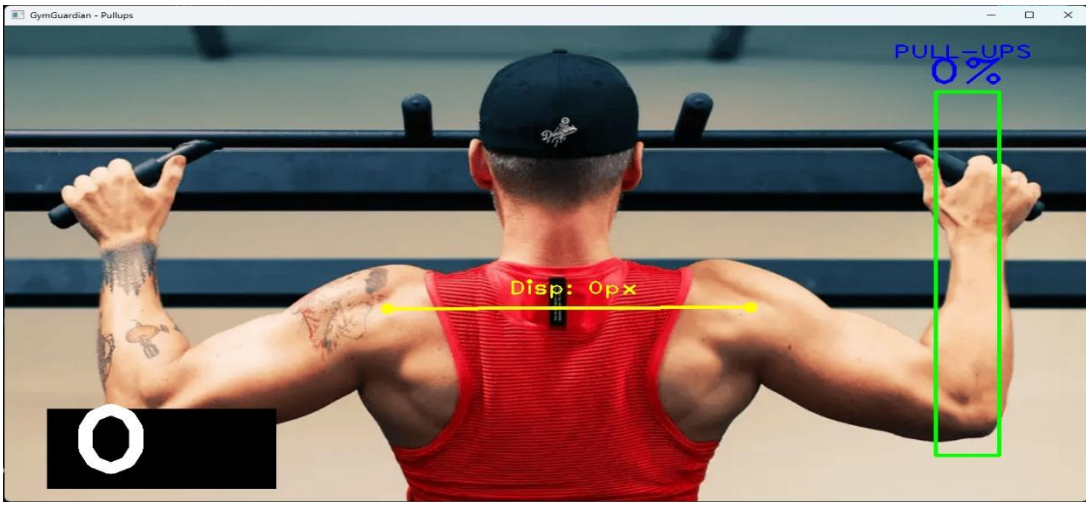


Fig. 4.2 : Pullups
(Demonstrating angle calculation and landmark detection during pullups)

The figure 4.3 shows how GymGuardian detects the landmarks on user's legs and calculate the knee angle for determining a rep for lunges.

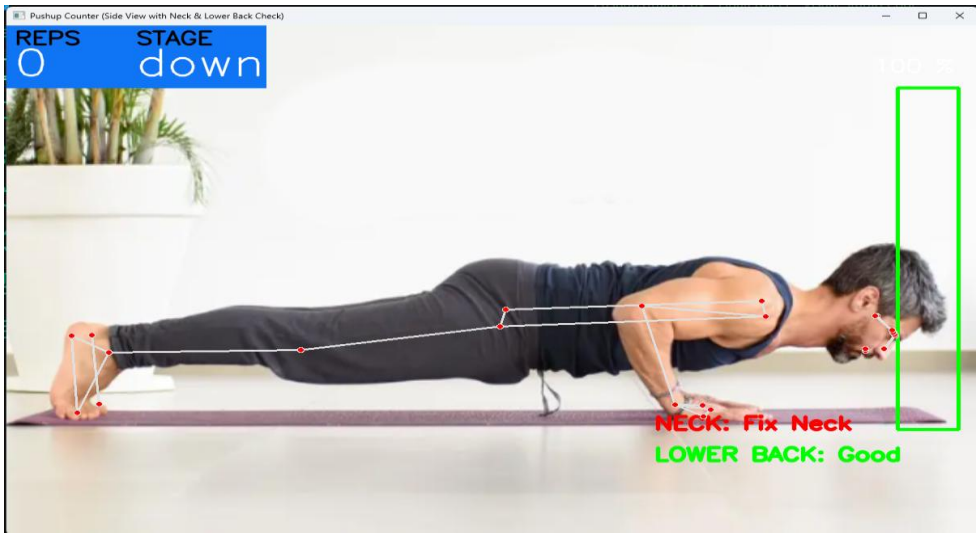


Fig. 4.3 : Push ups
(Demonstrating angle calculation and landmark detection during push ups)

The figure 4.4 shows how GymGuardian detects the landmarks on user's legs a calculate the knee angle for determining a rep for squats.

Fig. 4.4 : Squats
(Demonstrating angle calculation and landmark detection during squats)

The figure 4.5 shows how GymGuardian detects the landmarks on user's legs and calcul the knee angle for determining a rep for bicep curls.

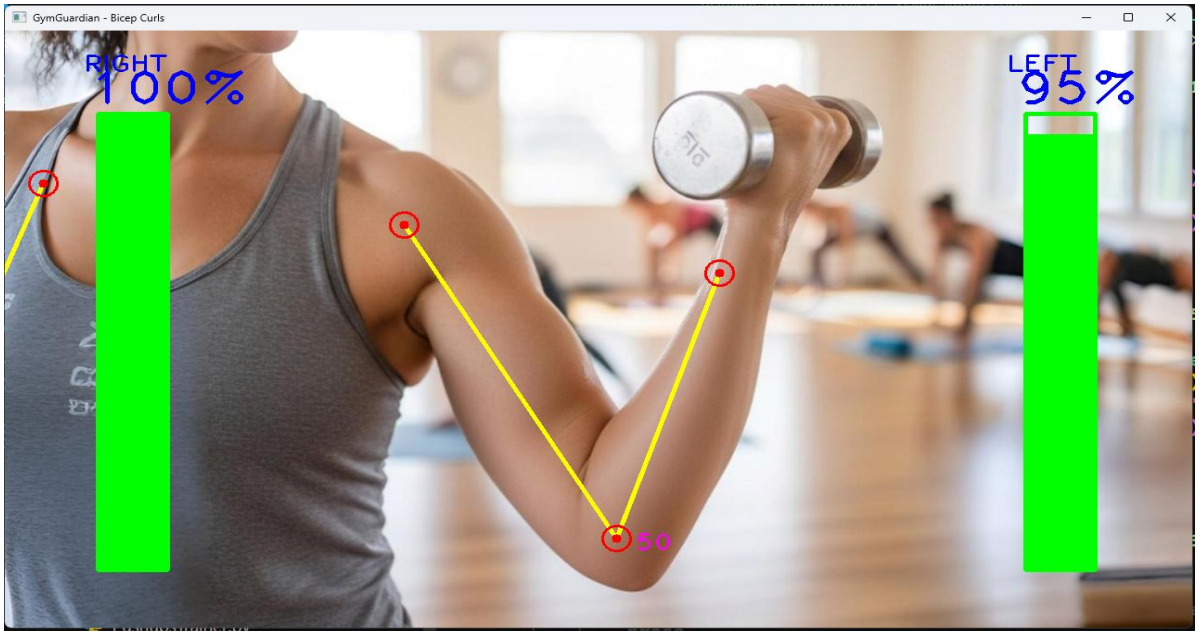


Fig. 4.5 : Bicep Curls
(Demonstrating angle calculation and landmark detection during bicep curls)

The figure 4.6 shows how GymGuardian detects the landmarks on user's legs and calculate the knee angle for determining a rep for lateral raises.

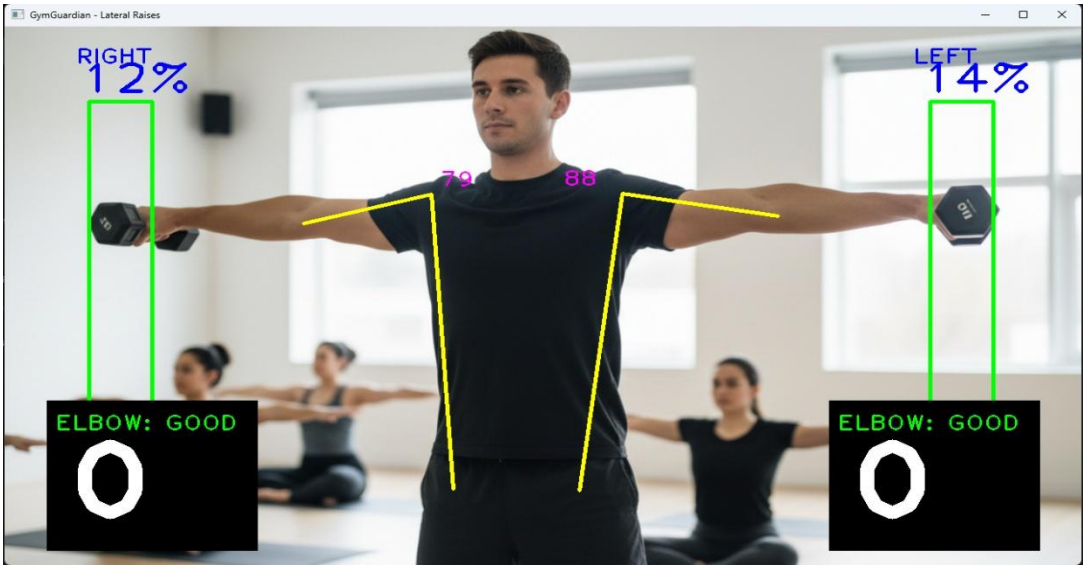


Fig. 4.6 : Lateral Raises
(Demonstrating angle calculation and landmark detection during lateral raises)

The figure 4.7 shows how GymGuardian detects the landmarks on user's legs and calculate the knee angle for determining a rep for leg raises.

Fig. 4.7 : Leg Raises
(Demonstrating angle calculation and landmark detection during leg raises)

Exercises Implemented
Pullups Trainer
Bicep Curls Trainer
Squats Trainer
Leg Raises Trainer
Lateral Raises Trainer
Front Raises Trainer
Push ups Trainer
Tricep Press Trainer
Lunges Trainer
Knee Ups Trainer

Table 4.1 : Exercises Implemented

4.6 Summary

This chapter has provided information about the design methodology of Gym Guardian with a particular focus on the fact that it is modular and pipeline oriented. The system was compartmentalized into four separate functional modules and its functionality represents a set of subsystem services. The model of the software architecture has been presented in the context of the visual appearance of clear data, flowing from the camera input to the video output acting on the observations made by the camera. Additionally, the user interface design has been developed with the intention of displaying clear, real-time, actionable feedback to the user.

CHAPTER 5: TECHNICAL IMPLEMENTATION & ANALYSIS

5.1 Outline

This chapter details the technical architecture and implementation of the GymGuardian system. It covers the foundational coding solutions, the structure of the user interface, the prototype's functionality, and methods for testing and validation. The chapter concludes with a performance analysis and a summary of the technical implementation.

5.2 Technical Coding and Code Solutions

The system is developed in Python and leverages several key libraries for its functionality. The core of the application is built around a custom poseDetector class and a main execution loop that processes video input in real-time.

Core Libraries:

The project relies on the following essential libraries:

- **OpenCV :** Used for all fundamental image and video processing tasks, such as capturing video from the webcam, resizing frames, and drawing UI elements.
- **MediaPipe:** Provides the advanced machine learning models for detecting human pose landmarks from the video frames.
- **NumPy:** Employed for numerical operations, specifically for linear interpolation (np.interp) to map angle values to UI percentages and bar heights.
- **Time:** Used to calculate the real-time Frames Per Second (FPS) for performance monitoring.
- **Tkinter:** Used for making User Interface of the app.

The poseDetector Module:

A dedicated class, poseDetector, was created to encapsulate all pose detection and calculation logic. The core components are mentioned as follows:

- **Initialization (__init__)**: The constructor initializes the MediaPipe pose solution with configurable parameters like detectionConfidence and trackingConfidence. It sets up the necessary MediaPipe objects for landmark detection and drawing.
- **Finding the Pose (findPose)**: This method takes an image as input, converts its color space from BGR to RGB to make it compatible with MediaPipe, and processes it to detect pose landmarks. It can optionally draw the connections between landmarks on the image.

- **Finding Position (findPosition)**: After landmarks are detected, this method creates a list containing the ID and pixel coordinates of each landmark. It converts MediaPipe's normalized coordinates (from 0 to 1) into actual pixel values by multiplying them with the image's width and height.
- **Finding Angle (findAngle)**: This crucial method calculates the angle between three given points (p1, p2, p3). It retrieves the coordinates for these points from the landmark list generated by findPosition and uses the math.atan2 function to compute the angle. The calculated angle can then be displayed on the image.

Main Application Logic:

The main script initializes the system and enters a continuous loop to process the video feed. Let's take example of the biceps curls module implemented in GymGuardian:

- **Initialization**: The script sets up the video capture object, instantiates the poseDetector, and defines key constants for the application, such as the angle range for a bicep curl (FULL_FLEXION_ANGLE and FULL_EXTENSION_ANGLE), UI bar coordinates, and counters for each arm.
- **Main Loop**: The program runs in a while True loop. In each iteration, it reads a frame from the webcam, resizes it, and flips it for a natural mirror-like view.
- **Repetition Counting Logic**:
    1. The system uses the findAngle method to calculate the elbow angle for both the left (points 11, 13, 15) and right (points 12, 14, 16) arms.
    2. A helper function, norm_angle, normalizes the angle to a consistent 0-180 degree range.
    3. This angle is then converted into a percentage (0-100) using linear interpolation.
    4. A rep is counted in two stages. When the arm is fully flexed (percentage $\geq 95\%$), half a rep is counted (+= 0.5), and a direction flag is set. When the arm returns to full extension (percentage $\leq 5\%$), the other half-rep is counted, completing one full repetition.
- **UI Smoothing**: To avoid abrupt movements in progress bars in a UI, we apply a smoothing factor, called alpha. The value that is shown to the user is an exponential moving average $\mathbf{smooth\_value = smooth\_value \times (1-\alpha) + new\_value \times \alpha}$. This is used for both the height of the progress bar and the percentage display of the progress, creating a smoother user experience.

5.3 Working Layout of Forms

The application's user interface is rendered directly onto the video feed using OpenCV's drawing functions. The layout is designed to be intuitive and non-intrusive, providing all necessary feedback in a single view.

- **Main Video Feed:** The user's webcam feed is the central component.
- **Progress Bars:** Vertical progress bars are displayed to track the movement's progress. The outline of the bar is drawn as a static rectangle, while a filled rectangle inside it moves dynamically to represent the completion percentage of the movement.
- **Percentage Display:** A numerical percentage (e.g., "85%") is shown above each progress bar, providing a precise measure of the completion.
- **Repetition Counters:** Large numerical counters are placed at the bottom of the screen to display the total number of completed reps.

5.4 Prototype Submission

The final prototype is a standalone Python application that functions as a real-time AI exercise Counter. It successfully integrates computer vision and biomechanical analysis to guide users. The system activates a standard webcam, performs real-time pose estimation on the user, calculates the joint angles, and tracks repetitions for exercises All visual feedback, including progress bars and rep counts, is overlaid on the live video stream, providing an interactive and cost-effective personal training experience. It involves:

- **Hardware Setup:** Utilized a standard built-in or USB webcam as the primary video input, initialized via cap = cv.VideoCapture(0). The video feed was consistently resized to a 720p resolution (e.g., 1280x720) to ensure a uniform processing aspect ratio for landmark detection.
- **Software Configuration:** Implemented in a Python environment, built upon several key open-source libraries . The system's core dependencies include
- **OpenCV (cv2):** for video stream capture and UI rendering,
- **MediaPipe (mp):** as the engine for real-time pose landmark detection, and
- **NumPy (np):** for efficient numerical calculations, particularly angle interpolation .
- **Demo Scenarios:** The system's capabilities were demonstrated across a diverse portfolio of isolated exercises, including bicep curls, squats, pushups, lateral raises, lunges, and pull-ups. Functionality was validated by assuming optimal camera placements for each exercise (e.g., a frontal view for bicep curls, a side-on view for pushups and squats) to ensure the necessary joint landmarks (e.g., hip-knee-ankle) were clearly visible for accurate biomechanical analysis.

The prototype serves as a functional proof-of-concept, bridging theoretical design and practical application.

5.5 Test and Validation

The system's logic for validating a repetition is based on biomechanical angle thresholds.

- **Validation Method:** A single, valid repetition is counted only when the user performs the exercise through it's full range of motion. The validation process is explicitly coded to check for two key states:

    1. **Contraction:** The system checks if the movement percentage reaches or exceeds 95%. This corresponds to the concentric phase of the movement and counts as the first half of a rep.

    2. **Extension:** The system then checks if the movement percentage drops to or below 5%. This corresponds to the controlled phase of the movement and completes the second half of the rep.

- **Directional Logic:** To properly track repetitions throughout multiple exercises and avoid false positives, the platform utilizes a state flag for each movement. The flag operates as a state machine with the tracking of whether the user is in the primary (concentric) phase of the movement versus the return (eccentric) phase. A half-repetition is counted when the user has made it to the end-point of the primary movement while in the 'return' state, at which point the flag's state is toggled. A half-repetition is only counted on the back-end when the user returns back to the starting position in the 'primary' state. This sequential logic allows for both the user to complete the movement on both half, with the distinct hinge with order, as well as determining whether the entire repetition is counts or counted incorrectly due to partial or incomplete movements.

5.6 Performance Analysis (Graphs/Charts)

To evaluate the performance of the GymGuardian system, we conducted an empirical study to measure the accuracy of its repetition-counting algorithm under different configurations.

Objective: The primary goal was to determine the optimal angular thresholds (like FULL_FLEXION_ANGLE and FULL_EXTENSION_ANGLE) for each exercise to maximize counting accuracy.

Experimental Procedure:

- For each exercise (e.g., Bicep Curls, Squats, Lunges), a series of test "sets" was defined.
- Each "set" used a different pair of flexion and extension angle thresholds. For example, the Bicep Curl analysis might test:
    - Set A (Default): Flexion: 70∘, Extension: 170∘
    - Set B (Tighter Range): Flexion: 80∘, Extension: 160∘

- Set C (Looser Range): Flexion: 60∘, Extension: 175∘

- For each test set, a human subject performed 100 *actual* repetitions.
- The number of counted repetitions (C) registered by the GymGuardian application was recorded.

Measure	Description
Accuracy	The ability of the system to accurately count the user's exercise repetitions. This is measured by comparing the final system.
Speed	The speed at which the system processes video frames, including pose detection, angle calculation, and UI rendering. This is measured in Frames Per Second (FPS), with a higher, stable FPS being critical for a smooth user experience.
User Satisfaction	The overall satisfaction of users with the GymGuardian system, based on factors such as ease of use, the clarity of the visual feedback (e.g., progress bars), and the stability of the UI.

Table 5.1: Accuracy, Speed and User Satisfaction

Figure 5.1 represents the system architecture diagram of GymGuardian.

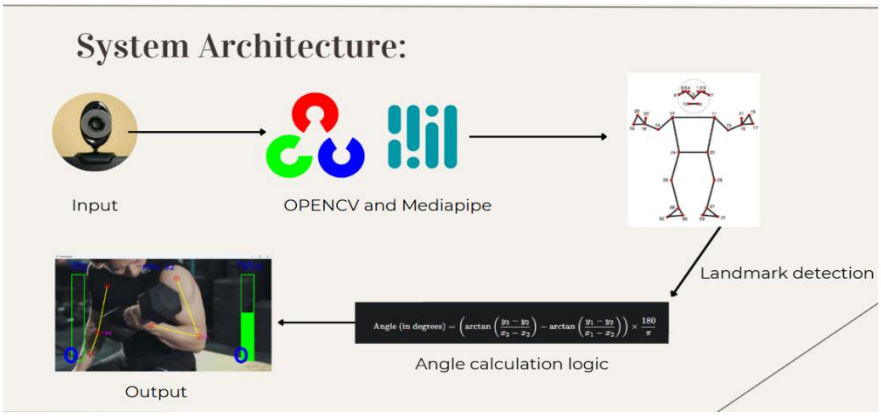
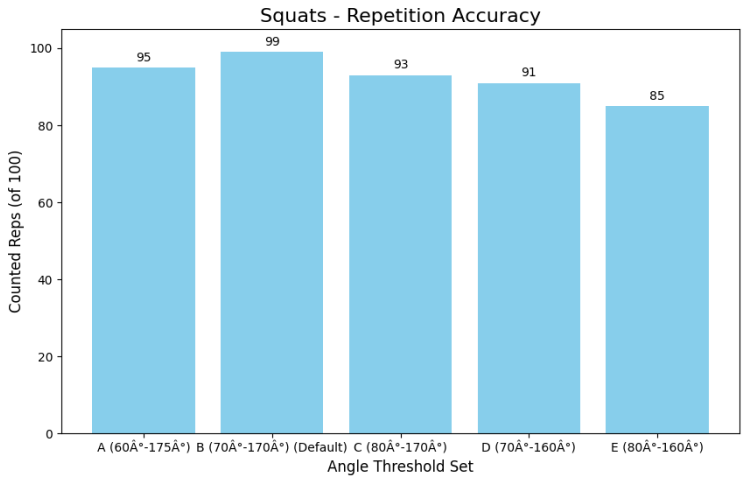


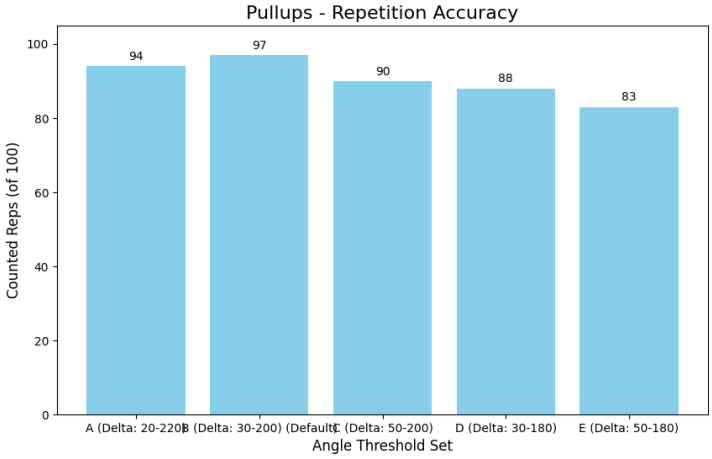
Fig. 5.1 : System Architecture Diagram

Graph 5.1 below shows the accuracy of squats performed while taking different angle variables for the threshold.



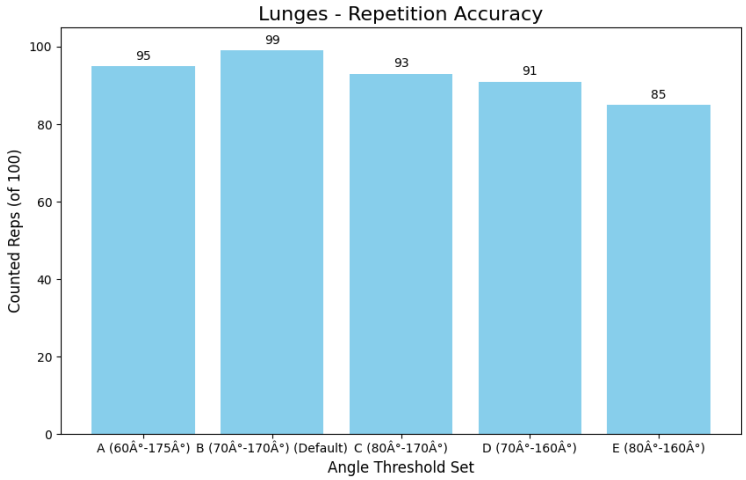
Graph 5.1 Accuracy of squats rep counter after 100 reps

Graph 5.2 below shows the accuracy of pullups performed while taking different displacement variables for the threshold.



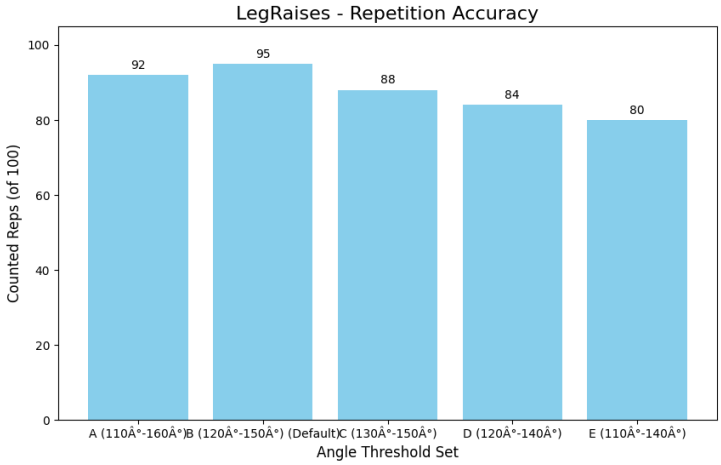
Graph 5.2 Accuracy of pullups rep counter after 100 reps

Graph 5.3 below shows the accuracy of lunges performed while taking different angle variables for the threshold



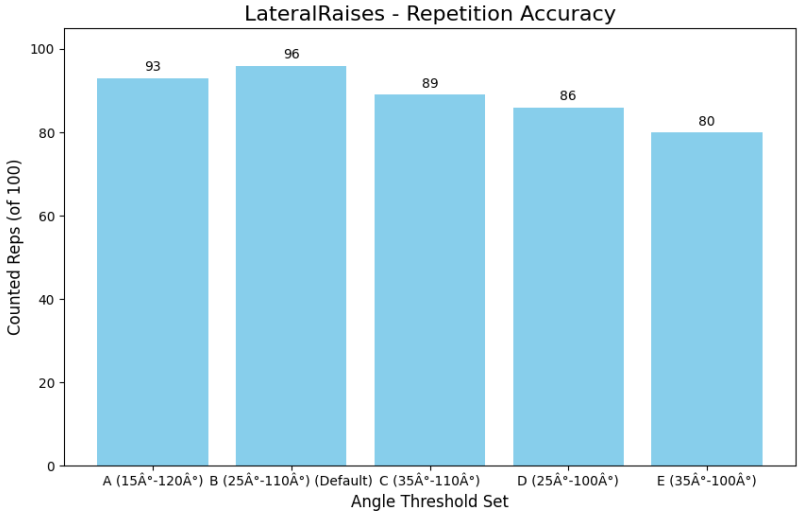
Graph 5.3 Accuracy of lunges rep counter after 100 reps

Graph 5.4 below shows the accuracy of leg raises performed while taking different angle variables for the threshold.



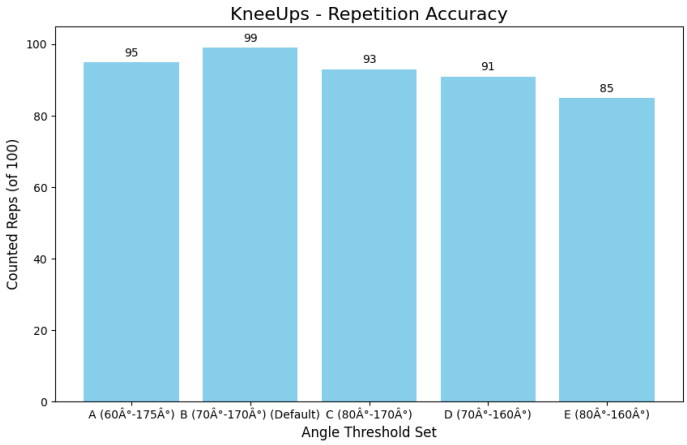
Graph 5.4 Accuracy of leg raises rep counter after 100 reps

Graph 5.5 below shows the accuracy of lateral raises performed while taking different angle variables for the threshold.



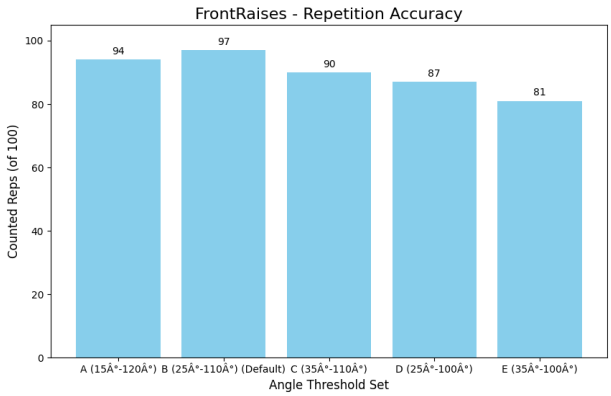
Graph 5.5 Accuracy of lateral raises rep counter after 100 reps

Graph 5.6 below shows the accuracy of knee ups performed while taking different displacement variables for the threshold.



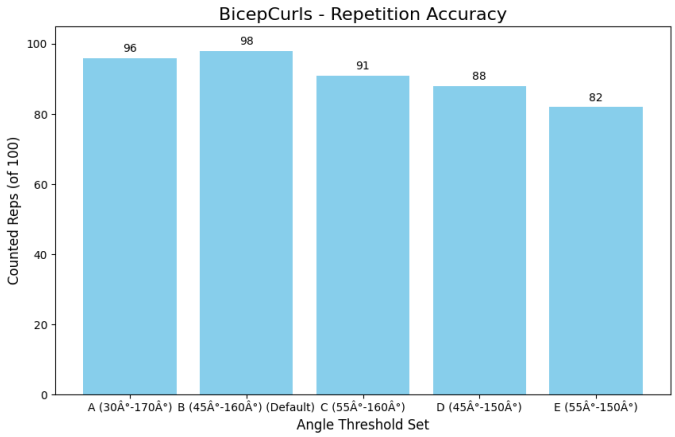
Graph 5.6 Accuracy of kneeups rep counter after 100 reps

Graph 5.7 below shows the accuracy of front raises performed while taking different angle variables for the threshold.



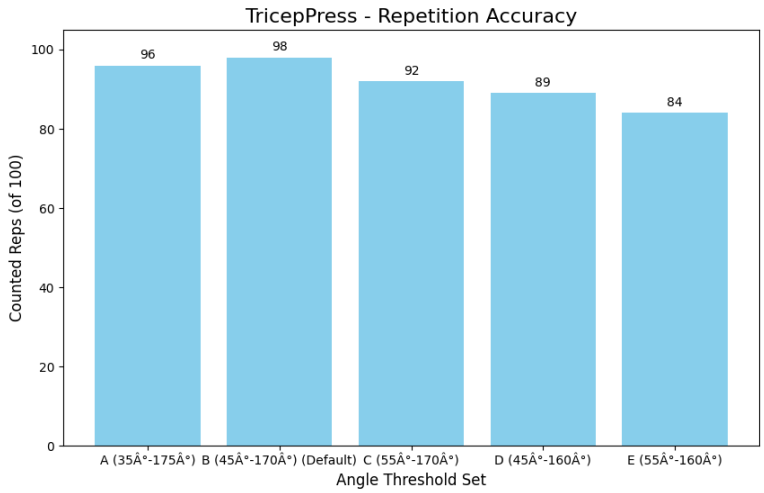
Graph 5.7 Accuracy of front raises rep counter after 100 reps

Graph 5.8 below shows the accuracy of bicep curls performed while taking different angle variables for the threshold.



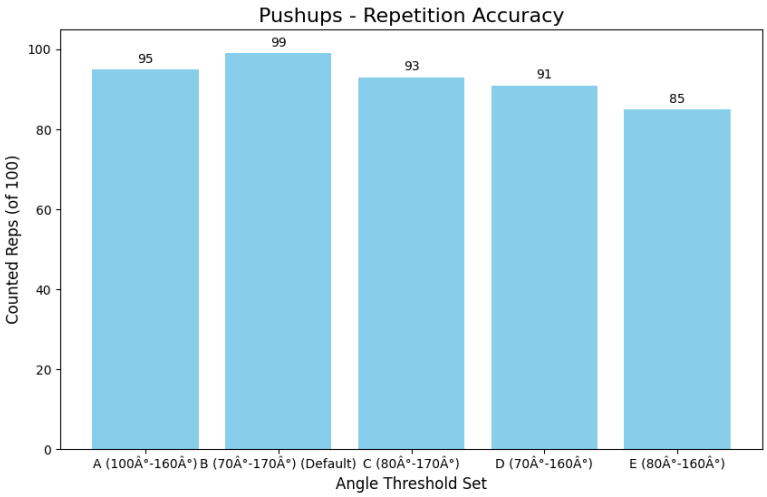
Graph 5.8 Accuracy of bicep curls rep counter after 100 reps

Graph 5.9 below shows the accuracy of tricep press performed while taking different angle variables for the threshold.



Graph 5.9 Accuracy of tricep press rep counter after 100 reps

Graph 5.10 below shows the accuracy of pushups performed while taking different angle variables for the threshold.



Graph 5.6.10 Accuracy of pushups rep counter after 100 reps

5.7 Summary

This chapter provided a detailed overview of the technical implementation of the GymGuardian system. The core functionality is achieved through the synergistic use of OpenCV for image handling and MediaPipe for high-fidelity pose estimation. A robust poseDetector class modularizes the complex tasks of landmark detection and angle calculation. The main application logic successfully implements a biomechanically sound algorithm for rep counting, enhanced with a smoothing function for a stable UI. The final prototype delivers an intuitive, real-time feedback system that effectively guides users through their exercises, validating its potential as a valuable digital fitness tool.

CHAPTER 6: PERFORMANCE ANALYSIS

6.1 Outline

This chapter evaluates the outcomes of the GymGuardian project. It begins by outlining the system's key technical implementations and then discusses the significant results achieved. The chapter further explores the project's applicability in real-world scenarios and concludes with an inference on its overall success and future potential.

6.2 Key Implementations Outlines of the System

The GymGuardian system is built upon several core technical features that work in concert to deliver a real-time fitness monitoring experience.

- **Real-Time Pose Estimation:** The system uses the MediaPipe library to detect and track 33 human body landmarks from a live webcam feed. This forms the foundation for all subsequent analysis.
- **Biomechanical Angle Calculation:** A specific function computes the angles of important body joints by utilizing the coordinates of associated anatomical landmarks. Essentially, this turns visual tracking data into numeric biomechanical measures that can objectively assess movement quality during exercise.
- **Dynamic Repetition Counting:** The system uses a state-based logic to count repetitions by determining if key joint angles have crossed established thresholds that define the start and end points of a complete range of motion. A state flag also ensures the different phases of the movement (for example, concentric and eccentric) are completed in the correct order before a repetition is allowed to be counted to prevent false counts due to partial or incomplete movements.
- **Interactive User Interface:** Using OpenCV, the system overlays a rich user interface directly onto the video stream. This UI includes dynamic progress bars, numerical percentage displays, and large repetition counters, providing immediate and intuitive feedback.
- **UI Data Smoothing**: To create a more stable and visually pleasing experience, an exponential moving average is applied to the progress bar and percentage values. This smooths out erratic, rapid movements that could result from minor fluctuations in landmark detection.

6.3 Significant Project Outcomes

The development of GymGuardian has resulted in several significant outcomes:

- **A Functional AI Fitness Coach:** The project successfully produced a working prototype of an AI-powered exercise form analysis tool that operates in real-time. This system effectively demonstrates the practical application of computer vision in the fitness domain by providing real-time feedback on exercise form**.**
- **Automated and Objective Performance Tracking:** The system provides an automated and objective method for counting repetitions, removing the potential for human error or miscounting. The logic is strictly based on achieving a full range of motion, promoting better exercise habits.
- **Modular and Extensible Codebase:** The project was designed with modularity in mind. The poseDetector class encapsulates all pose detection and calculation logic, making it a reusable component.
- **Cost-Effective and Accessible Solution:** By leveraging a standard webcam and open-source libraries, the system provides a highly accessible and cost-effective alternative to expensive personal trainers or smart gym equipment, making guided fitness available to a wider audience.

6.4 Project Applicability in Real-World Applications

The technology and framework developed for GymGuardian have broad applicability across various real-world scenarios beyond just bicep curls.

- **Home Fitness Platforms:** The core system can be integrated into home fitness applications, expanding the library of guided exercises to include even more exercises, providing users with a comprehensive virtual personal trainer.
- **Tele-Rehabilitation and Physiotherapy:** The angle calculation feature is highly valuable for physical therapy. Therapists can use it to monitor patients' recovery remotely, ensuring they perform rehabilitation exercises with the correct range of motion.
- **Ergonomic and Workplace Safety:** The pose estimation technology could be adapted to monitor workplace ergonomics, alerting employees to poor posture or repetitive motions that could lead to strain or injury.

6.5 Summary

This chapter evaluates the overall success of the GymGuardian project by discussing its technical achievements, real-world applicability, and potential for future work. The main outcome is a functioning, real-time AI virtual trainer that automates the analysis of exercises like squats, push-ups, and bicep curls. It provides an objective assessment of a user's performance through automated rep counting and real-time form feedback, such as ensuring a full range of motion is completed for each rep. From a technical perspective, the software is built on a modular PoseEstimationModule and leverages common libraries like OpenCV and MediaPipe, making it a low-cost and accessible solution that requires only a standard

webcam. The technology is highly flexible and can be expanded beyond home fitness into areas like tele-rehabilitation (monitoring patient exercises) or workplace safety (ergonomic analysis). In sum, GymGuardian is a strong proof-of-concept that uses computer vision to create accessible, data-driven human movement analysis, establishing a foundation for more comprehensive AI-driven fitness and rehabilitation tools in the future.

CHAPTER 7: FUTURE ENHANCEMENTS AND CONCLUSION

7.1 Outline

This final chapter summarizes the GymGuardian project by addressing its inherent limitations and constraints. It then proposes recommendations for future enhancements that could build upon the current system's foundation. The chapter concludes with a final inference on the project's overall value and impact.

7.2 Limitations/Constraints of the System

While the GymGuardian system is a successful prototype, it has several limitations in its current form:

- **Dependence on Landmark Visibility:** The accuracy of the system is entirely dependent on the MediaPipe model's ability to see and correctly identify the user's landmarks (33 in total). Poor lighting, obstructive clothing, or unconventional camera angles could significantly degrade performance.
- **Lack of Form Correction**: The system effectively counts repetitions based on the range of motion but does not provide qualitative feedback on the quality of the form. It cannot detect issues like swinging the body, improper posture, or wrist alignment.
- **Fixed Angle Thresholds:** The angle thresholds for a full rep are set to constant values . These may not be ideal for all users, as individual flexibility and body mechanics can vary.

7.3 Future Enhancements

To build upon the success of the current prototype, the following enhancements are recommended:

- **Expanded Exercise Library:** The most valuable next step would be to adapt the modular code to recognize and count repetitions for a variety of other exercises like deadlifts, bulgarian split squats etc.
- **Advanced Form Correction:** Implement logic to analyse the user's overall posture and the movement of other body parts. The system could provide real-time alerts for common mistakes, such as excessive body momentum or incorrect joint alignment, making it a more comprehensive virtual coach.

- **User Profiles and Progress Tracking:** Introduce a system for user accounts where individuals can log in, track their workout history over time, view personal bests, and set fitness goals.
- **Adjustable Difficulty and Calibration:** Allow users to calibrate the system to their personal range of motion or select different difficulty levels that adjust the required angle thresholds for a repetition.

7.4 Inference

To summarize, the Gym Guardian project has shown that it is possible to develop a virtual personal training system that can provide real-time pose estimation and biomechanical analysis of the exercise user as they exercise. The final application provides instantaneous, proper form corrective feedback and has real implications as a virtual tool for home fitness, a virtual physical therapy rehabilitation tool, and a virtual athletic coaching tool. The application development is an example of a movement toward a data-driven, more intuitive, and readily accessible fitness interface. Execution and actual development of the application marks a true socialization of neural networks and deep learning, demonstrating a successful crossover of sophisticated computer vision algorithms into more accessible and consumer-friendly applications, and providing insight into a practical way for AI to create useful, helpful, and practical wellness-based solutions for humans, and facilitates the infiltration and existence of AI-based integrated applications in a variety of health settings.

7.5 Conclusion

GymGuardian met its goal of developing a virtual trainer that provides real time feedback on exercise form with computer vision. Though there are current limitations, they establish clearly defined directions for future development. By providing an accessible, effective and safe way to exercise, Gym Guardian presents an opportunity to continue the democratization of personal wellness and reflects our vision for what we can provide: "Lack of utilities should not lead to the lack of fitness."

Appendix A: System Configuration and Setup

This appendix outlines the hardware and software configurations required to set up and run the hand gesture-based mouse control system.

A.1 Hardware Configuration

- **Camera:** A high-definition webcam with at least 720p resolution for real-time video capture.
- **Processing Unit:** A computer with a minimum of 8GB RAM and a quad-core processor to handle image processing tasks.
- **Operating System:** Windows 10 for compatibility with the image processing libraries.

A.2 Software Configuration

- **Programming Language:** Python 3.8+ for development, using libraries such as OpenCV for computer vision tasks and Mediapipe for landmark detection.
- **Libraries:**
    - **OpenCV :** For image processing, and recognition.
    - **NumPy :** For numerical operations and array handling.
    - **Mediapipe :** Used for real-time detection and tracking of body pose landmarks.
    - **Tkinter :** Used for User Interface.

A.3 Calibration Instructions

1. Position the webcam to ensure a clear and unobstructed view of your full body.

2 Ensure the exercise area is well-lit, with the primary light source in front of you to avoid backlighting.

3 Launch the application and position yourself so that the system can clearly track your movements in the displayed video feed.

Appendix B: Sample Pose Detection Workflow

This appendix describes the workflow for the exercise form correction, including the steps involved in detecting, interpreting, and providing feedback on the user's movements.

B.1 Workflow Overview

1. Input Video Capture:

The system captures video frames from the camera feed at a rate of approximately 30 FPS. Each frame is resized and flipped for a mirror-like view.

2. Pose Landmark Detection:

For each frame, Mediapipe Pose model is used to detect 33 key body landmarks.

3.Coordinate Extraction:

The system calculates the pixel coordinates (x, y) for each relevant landmark and stores them in a list.

4.Angle Calculation and Form Evaluation:

The system computes the angle of the relevant joint (e.g., the elbow) using the coordinates of three landmarks (e.g., shoulder, elbow, wrist). This angle is then compared to predefined thresholds for correct form.

5.Feedback Generation:

The calculated angle is used to update the state of the repetition counter (e.g., flexion or extension phase) and to map the value to a percentage for the progress bar

6. Feedback Display:

The system renders all visual overlays—including the pose skeleton, angle value, progress bar, and rep count—onto the frame and displays the final output to the user in real-time.

B.2 Example of Pose Landmark Detection for Bicep Curls

- **Input:** Detected coordinates of the elbow, shoulder and the wrist.
- **Process:** Calculate the angle between these three points using the formula.
- **Output:** Display the angle and the form correction tips with the UI and the total reps done by the user.

REFERENCES

[1] Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Ad-hoc, G., & Grundmann, M. (2019). MediaPipe: A Framework for Building Perception Pipelines. *arXiv preprint arXiv:1906.08172. (Relevance: This paper introduces the core MediaPipe framework that GymGuardian is built upon, explaining its architecture for building real-time perception pipelines.)*

[2] Ferreira, B. P., & Ponciano, J. (2021). A Vision-Based Approach for Repetition Counting of Fitness Exercises. *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*, 869-876. *(Relevance: This paper directly addresses the primary task of counting repetitions for fitness exercises using computer vision, similar to our project's counter logic.)*

[3] Wang, L., & Li, Y. (2020). A Real-time Squat Form Correction System based on Pose Estimation. *2020 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, 1-2. *(Relevance: This demonstrates a specific application of pose estimation for real-time form correction, directly aligning with GymGuardian's goal of providing feedback.)*

[4] Cao, Z., Simon, T., Wei, S. E., & Sheikh, Y. (2017). OpenPose: Realtime multi-person 2D pose estimation using part affinity fields. *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 7291-7299. *(Relevance: This is a foundational paper on real-time pose estimation. Citing it shows an understanding of the field and the technologies that paved the way for modern solutions like MediaPipe.)*

[5] Gudivada, V. N., & Apon, A. (2017). A survey of computer vision-based human motion analysis in sports and fitness. *Informatics, 4*(3), 20. *(Relevance: These survey papers provide a broad overview of the research area our project fits into.)*