International Journal of Engineering, Management and Humanities (IJEMH)
Volume 4, Issue 6, Nov.-Dec., 2023 pp: 106-119 www.ijemh.com

"Computer Vision Based Workout Application"

Tejas D R, Vishnu J G, Srujan A S, Pradeep V, Divya S J
Tejas D R, PES University, Bangalore, Karnataka.
Vishnu J G, PES University, Bangalore, Karnataka.
Srujan A S, PES University, Bangalore, Karnataka.
Pradeep V, PES University, Bangalore, Karnataka.
Divya S J, Assistant Professor, PES University, Bangalore, Karnataka.
Corresponding Author: Tejas D R.

Date of Submission: 09-11-2023 Date of Acceptance: 23-11-2023

ABSTRACT
The use of software in the health care sector has proved to be essential over the past few years. The use of software by doctors has helped them better engage and monitor their patients. Now with the increased fear of pandemic and the development of Computer Vision in healthcare has helped patients to be guided even without having to visit the hospital frequently for routine exercises. Patients can work out at the comforts of their homes and still be monitored closely on their health improvements.
This project is focused on building a workout application which is made for people who wish to work out on their own without any human guidance and with minimal physical interaction with their device. The project is planned to be implemented in python OpenCV using none or minimal machine intelligence to improve performance. The project is focused on building a workout application using physical constraints, as provided by the medical practitioner, and no ML models are built to identify poses.

I. INTRODUCTION

Computer vision in the healthcare sector has developed rapidly over the past few years. The importance of Computer vision in the healthcare industry cannot be emphasized. Its techniques have been proven useful in many medical situations, including surgical planning and medical imaging, and they are widely used today.

The COVID-19 pandemic came with many disruptions in every aspect of the society, all the sectors of the world were greatly affected. There is lots of visual data which is used in the healthcare sector, such as photographs and scans, and diagnoses are to be made using this data. Computer vision is used as a tool to aid in making diagnosis from these data. For example by helping the patients in situations where doctors are unreachable or unavailable.

During the pandemic the development of Computer Vision in healthcare has helped patients to be guided even without having to visit the hospital frequently for routine exercises. Patients can work out at the comforts of their homes and still be monitored closely on their health improvements. So our Project focuses on building a workout application which is made for people who wish to workout on their own without any human guidance and minimal physical interaction with their device.

Over the past decade machine learning has gained a lot of attention and is able to make its way into many day to day life applications such as visual recognition, data analysis, robotics, etc,. Machine learning finds its application in almost every walk of life. Although ML is an emerging technology, it is sometimes excessive and unnecessary for the task at hand.

Once such application in posture recognition where ML models are built just to recognize the current pose the person in the frame stood out to be a less optimal approach and required a more efficient solution for faster image processing and recognition. A simple constraint can be used to identify just the necessary positions so that the pose can easily be recognized. The only ML required in this approach would be to extract the skeleton of the user in frame.

Not only does this improve efficiency, this also helps developers in easily building newer constraints as and when required for multiple poses. This can particularly be used in an application which deals with providing guidance for exercises using computer vision. Constraints such as angles and inclination can be used to understand the stage of exercise the user is currently in and guide the user

International Journal of Engineering, Management and Humanities (IJEMH)
Volume 4, Issue 6, Nov.-Dec., 2023 pp: 106-119 www.ijemh.com

better if there are any errors in the posture that the user is currently in.

Furthermore the user has a visual feedback mechanism by which he can understand and improve his exercises over time. This works just like working out in front of the mirror but in this case the user can also see stats and other info on the screen in real time. The draws a better picture for the user so that he/she can workout without having to worry about visiting a physical guide.

II. Problem Statement

There is an increase in the need for software to be implemented in health care so that doctors can deliver better treatment for their patients. Engaging and monitoring patients can be simplified and improved by softwares. Specifically, Computer Vision plays an important role in bringing doctors closer to their patients. Over the last decade Computer Vision has gained a lot of attention in the health care and fitness sector. Implementing Computer Vision automates many monitory jobs of doctors and other hospital staff. Monitoring an ICU patient, continuous and careful monitoring of patients during their regular exercises, etc.,

Over the last few years, due to the pandemic, people feared visiting doctors frequently at hospitals, social distancing was the new norm. But this not only caused patients to miss regular exercise but also caused problems for the doctors treating them. The lack of regular contact and monitoring of patients has caused many problems and may even lead to critical health conditions. To overcome this gap, we planned on bringing the doctor/guide home instead of bringing the patients closer to the doctors.

Patients need a guide/doctor on spot so that they do not make any errors which may sometimes cause some adverse side effects. Neither doctors nor patients desire to risk encountering unforeseen problems due to unguided or unmonitored circumstances. Here arises the need for a software that can guide the patients in their regular exercise and also give the doctors a tool to monitor and tweak the exercise as per requirement. This way communication between the doctors and the patients improves and reduces any undesired outcomes.

A virtual guide/doctor can be implemented with computer vision for posture detections and analysis. An application to consolidate all the data gathered from these video frames, storing and presenting the data in a user-friendly application can bring in a lot of change in the way doctors can interact with

III. PROJECT REQUIREMENTS SPECIFICATION

This chapter deals with the main goals and requirements of the project under consideration. This includes the data required, the front-end requirements, database design requirements and a working constraint-based workout module.

3.1 Workout module

This section involves development of a workout module for recognition of exercise poses through the image frames obtained from the user's camera. The module applies certain constraints on the user's current frame and determines the current state of the user. The skeleton frame of the user can be obtained using any module such as mediapipe. Constraints are applied on these skeleton images to identify the current state of exercise of the user. This data can be further used to guide the user, update the database and also build statistical information pages. The module should also give visual feedback such as constraint angles, workout count and the stage of the workout the user is currently in.

3.2 Front-end for operation

This section involves building a control room kind of front-end for the user, so that he/she can change and set the constraints according to their preferences. This is necessary as the constraints vary from user to user. This control allows users to easily control their exercises. They can also navigate different exercises and set constraints for individual exercises. Since this is a control and navigation tool, we can use this to access different functions of the application such as logging in, creating new users, read disclaimer, set constraints, select exercise to perform, display the stats page and initiate new exercise as and when needed.

3.3 Back-end Modules

3.3.1 Database Module

This module requires database design for easy storing and access of data from the corresponding tables. The table should be able to store data both user data in consistent tables and in date wise sectioned tables so that we can store time varying data for building the stats page. The database does this by creating new tables for every user. The database should handle all updates and changes to tables instantaneously and accurately. This involves updating multiple tables, creating and deleting tables/rows.

3.3.2 Stats Pages

This involves importing data pertaining to a particular user as and when required, instantaneous update of the database when the user completes an exercise. This way we have a common interface for database operations. The data of each user can be used to build visual stats for users and medical practitioners for better and easier analysis of exercises performed.

3.4 Functional Requirements

The workflow goes as follows:

- The application has an interface for selecting the exercise to perform and setting angles for constraints. It also has access to start and stop exercises.
- The camera sends in stream of frames (video frames) to the application
- The application analyses this creates a mapping of body parts and returns the coordinates of important landmarks of the body in the frame
- This data is compared with the predefined constraints, such as angle between limbs, inclination, etc.
- The frames with the coordinates mapping are displayed to the user
- In case of an error detected, visual feedback is returned
- If no errors are detected then the application stores state for further analysis, state such as count or time, and discards the current frame.
- Other functionalities of the application include login and new user creation
- Newly added functionality includes a stat page which has visual representation of the data being stored from the user exercises.
- Other errors such as frames with no user detected are ignored
- The application displays the user with his own frame, which acts like a mirror, and also frames showing how to go about doing the exercise accurately.

3.5 External Interface Requirements

3.5.1 User Interface

we have planned the application Front end as follows

- The front-end application is planned to be built using one of python's front-end GUI modules.
- The count will be displayed on one corner for the user's reference along with angles for constraints identification.
- The user can browse through all the available exercises in the application and start the one he/she desires to work on.
- The angles change color when the constraints are met, or the user is within the constraint set.
- This analysis is expected to occur in real time. With maybe 1 sec delay at max.
- The application also has a stats page which counts and records user exercises on a day-to-day basis.

3.5.2 Hardware Requirements

The only hardware that is the necessity of this application is a webcam. This hardware is used to capture the user's actions. The connection and transfer of frames from the camera to the application is carried out by the OpenCV module of python. MediaPipe is another module which recognizes the hand gestures and gives the coordinates for a few specific points on the user's hand. OpenCV allows python programs to access the webcam of the device for as long as the program runs.

3.5.3 Software Requirements

- OpenCV2: The version of OpenCV2 used in our application is 4.5.5, this version is compatible with python version>2.7
- Python: The main programming language used is python and the versions of the same used are 3.7.9 and 3.9.6. Although all the development plans are in the 3.9.6 version, testing will be done in both the versions.
- Media pipe
- PySimpleGUI: The front-end module planned, and this is used to set constraint, login, see stats, create new users and much more.
- OpenCV2: The version of OpenCV2 used in our application is 4.5.5, this version is compatible with python versions >2.7

3.5.4 Communication Interfaces

Since our application is restricted to work in a localized configuration, we are not presently working with network transfer of any information between devices. But since the database has the ability to store and entertain multiple users on a single machine, we have moved our application to a localized multiuser domain. This enables easy transfer of necessary and accurate data as and when required.

The GUI used helps in easily communicating with all the other modules such as the exercise module, the stats module, the login module, etc.. The GUI acts as a central system to control all other modules and hence we have a simpler single debugging area.

International Journal of Engineering, Management and Humanities (IJEMH)
Volume 4, Issue 6, Nov.-Dec., 2023 pp: 106-119 www.ijemh.com

3.6 Non-Functional Requirements

3.6.1 Performance Requirements

In a condition where a high computational powered GPU is used, users can find bare minimum latency, the time to process each frame is a few milliseconds. Besides, the computation for each exercise has 2 tasks, one correcting the user's exercise and the other storing the data in the database for further analysis tasks. The latency of storing the data is pertaining to accessing the row in the table for that particular user and updating the row. Whereas the correction requires the program to access the set constraints by the user and match them with the user's frames. These database performance hindrances apply to many of the operations such as login, stats, etc.

3.6.2 Safety Requirements

The basic principle of our application is that we are not saving any frames, rather just extracting important and useful information from each frame and giving results. Due to this reason, there are no worries of usage of excessive storage. Since multiple users can access and work on the same application on the same device, we need authentication for maintaining privacy.

The application also has a health safety disclaimer at the beginning of the application which warns the user of the risks and responsibilities before using the application.

IV. SYSTEM DESIGN

This chapter deals with the system design of the application, this involves high level design for the application, low level design and operation model for different modules available. This chapter also includes suggested and tried out APIs and modules which did not fit well with the application.

4.1 High Level Design

4.1.1 USE Case Diagram

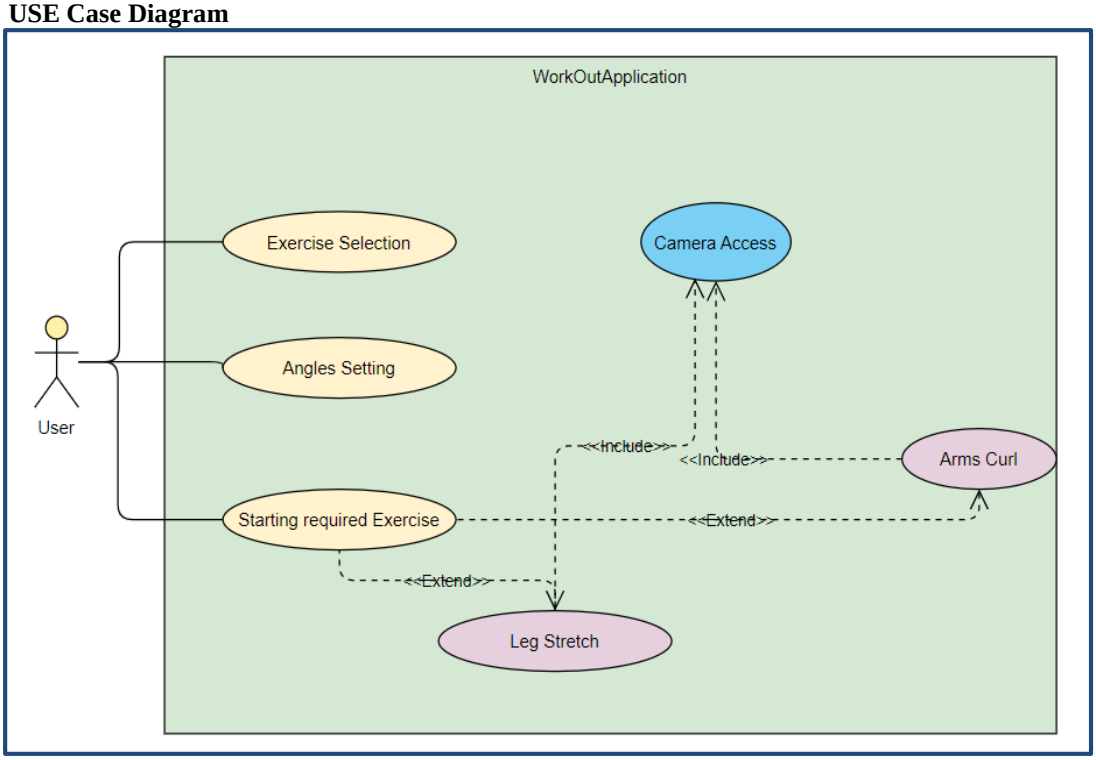

Fig. 4.1.1a

International Journal of Engineering, Management and Humanities (IJEMH)
Volume 4, Issue 6, Nov.-Dec., 2023 pp: 106-119 www.ijemh.com

Use Case Item	Description
Exercise Selection	Selection of exercise
Angles Setting	Setting the boundary conditions
Starting required Exercise	Starting the selected exercise
Camera access	Accessing the camera
Leg stretch	Exercise 1
Arms curl	Exercise2

4.1.2 Class Diagram

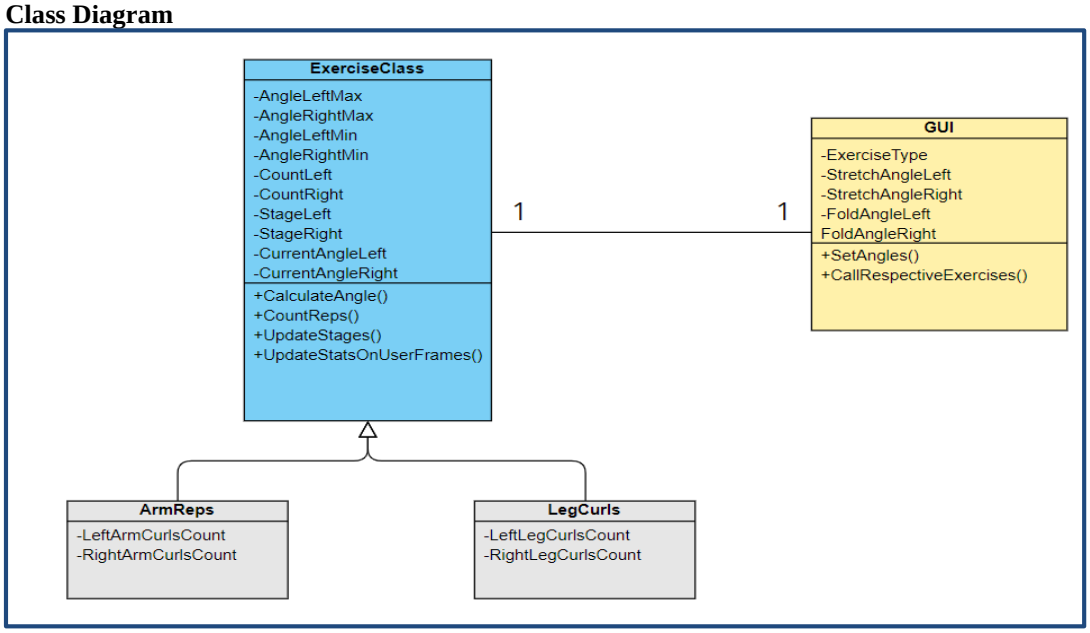

Fig. 4.1.2a

International Journal of Engineering, Management and Humanities (IJEMH)
Volume 4, Issue 6, Nov.-Dec., 2023 pp: 106-119 www.ijemh.com

4.1.3 State Diagram

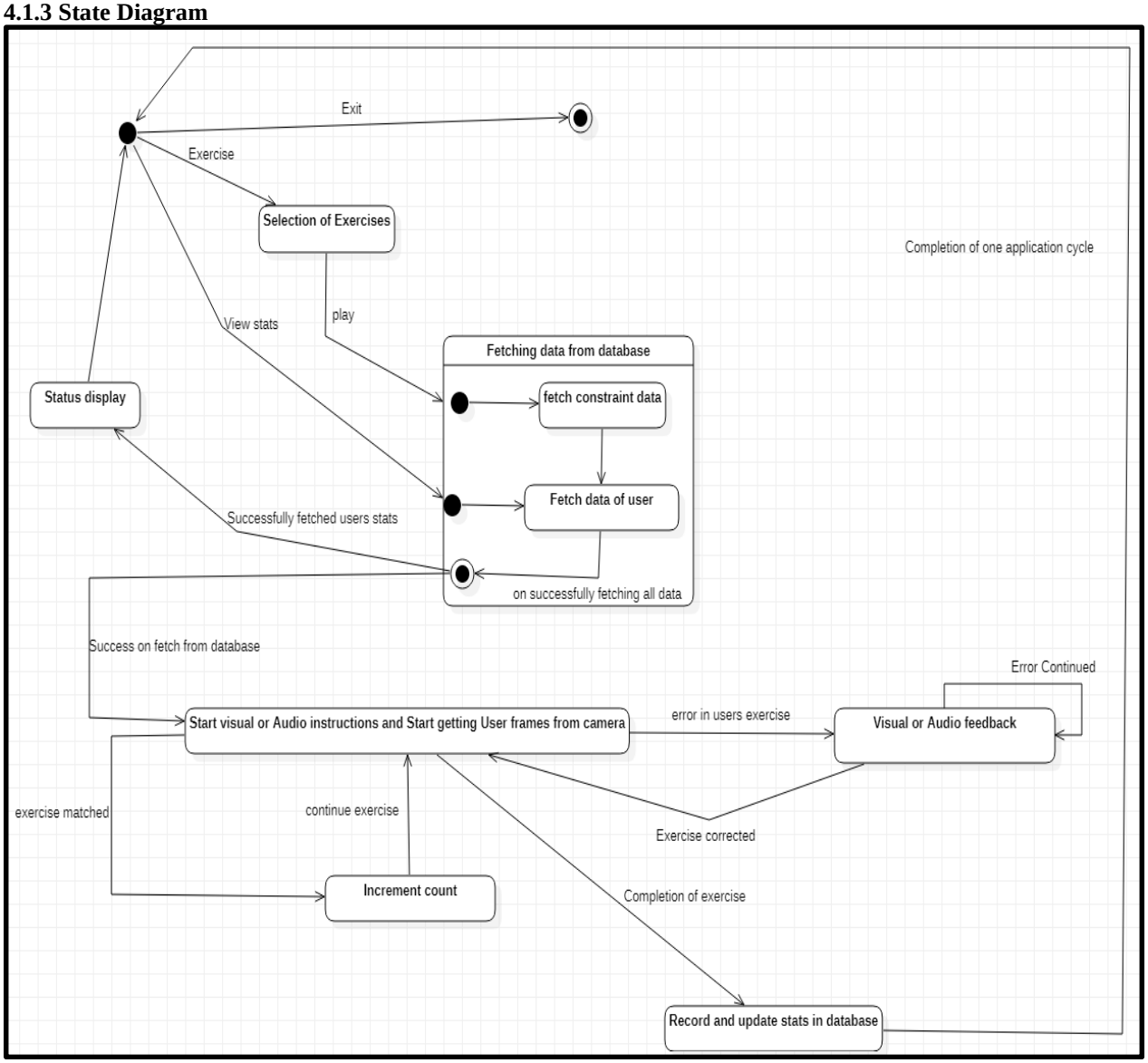

Fig. 4.1.3a

International Journal of Engineering, Management and Humanities (IJEMH)
Volume 4, Issue 6, Nov.-Dec., 2023 pp: 106-119 www.ijemh.com

4.1.4 Posture Control and Data flow High level Design

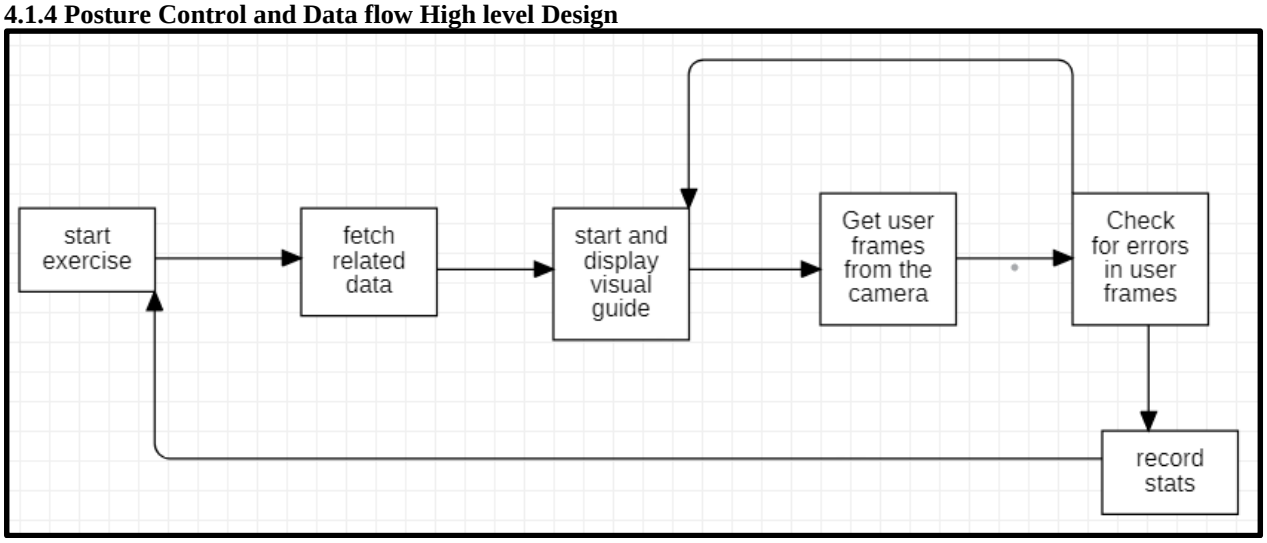

Fig. 4.1.4a

4.1.5 Gesture Control and Data flow High level Design

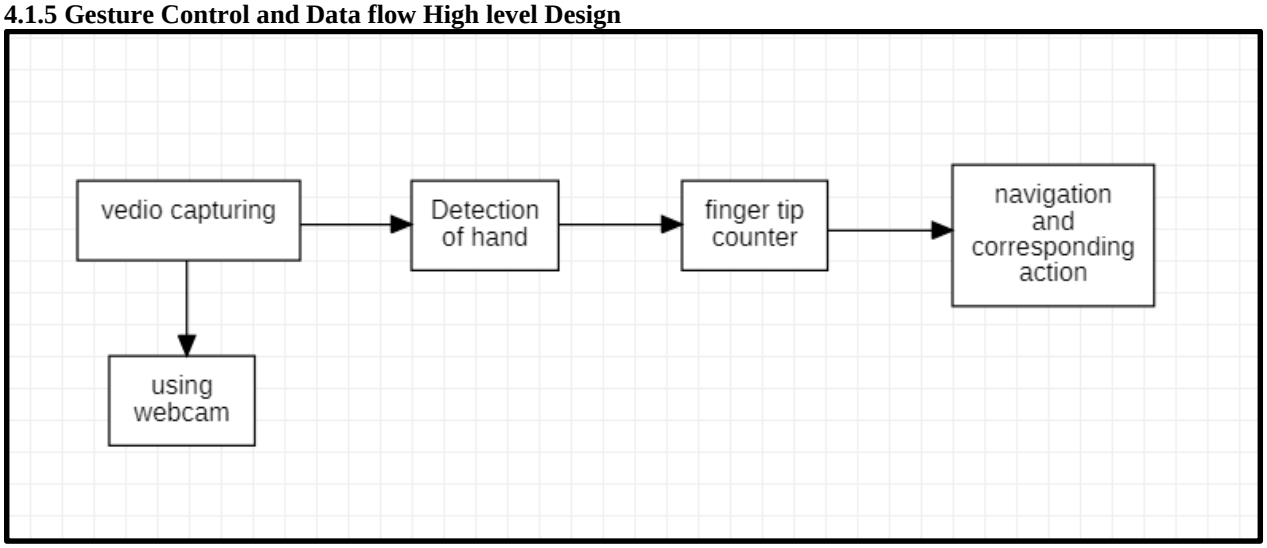

Fig. 4.1.5a

4.2 Low Level Design

4.2.1 Class 1: GUI

User interface, an application interface for users to navigate through the app. Displays the different exercises to select and start the session, also displays stats regarding the user and other details. This also allows users to set constraints before starting the exercise so that visual feedbacks can be provided as and when necessary.

4.2.1.1 Data members

Data Type	Data Name	Access Modifiers	Initial Value	Description
String	ExerciseType	private	Select	Which type of exercise to be selected
Float	StretchAngleLeft	private	160.0	Max value of the angle for left side

International Journal of Engineering, Management and Humanities (IJEMH)
Volume 4, Issue 6, Nov.-Dec., 2023 pp: 106-119 www.ijemh.com

Float	StretchAngleRight	private	160.0	Max value of the angle for right side
Float	FoldAngleLeft	private	30.0	Min value of the angle for left side
Float	FoldAngleRight	private	30.0	Min value of the angle for right side

4.2.1.3 Method 1: setAngles ()
The following details shall be defined for the methods:

- Purpose: To set the boundary conditions pertaining to the exercise.
- Input: angles pertaining to StretchAngleLeft, StretchAngleRight, FoldAngleLeft, FoldAngleRight.
- Output: Set the boundary angle values accordingly to the input parameters.

4.2.1.4 Method 2: CallRespectiveExercise ()
The following details shall be defined for the methods:

- Purpose: Calls the exercise selected and runs the session (captures the video and monitors it based on the constraints).
- Input: The exercise to be selected
- Output: Runs the session.

4.2.2 Class Name 2: ExerciseClass ()
4.2.2.1 Class Description
This class runs the exercise session required by the user. Captures the video, analyses the frames, calculates the required physical constraints (eg: angles at joints) and compares with boundary conditions and updates the result.

4.2.2.2 Data members

Data Type	Data Name	Access Modifiers	Initial Value	Description
Float	AngleLeftMax	private	-1	Max value of the angle for left side
Float	AngleRightMax	private	-1	Max value of the angle for right side
Float	AngleLeftMin	private	-1	Min value of the angle for left side
Float	AngleRightMin	private	-1	Min value of the angle for right side
int	CountLeft	private	0	Count no. of left repetitions
int	CountRight	private	0	Count no. of right repetitions
String	StageLeft	private	NULL	Exercise stage
String	StageRight	private	NULL	Exercise stage
Float	CurrentAngleLeft	private		Current left side angle
Float	CurrentAngleRight	private		Current right-side angle

4.2.2.3 Method 1: CalculateAngle ()
The following details shall be defined for the methods:

- Purpose: calculates the angle at the required joint.
- Input: The parameters for the angle to be calculated at the required joint (eg: arm and forearm to get angle at elbow).

- Output: calculated angle is returned and the CurrentAngleRight or CurrentAngleLeft attribute is updated accordingly.

4.2.2.4 Method 2: CountReps ()

The following details shall be defined for the methods:

- Purpose: counts the no. of repetitions performed by comparing with the boundary conditions.
- Input: None. (uses class variables)
- Output: returns no. of repetitions and updates the CountLeft or CountRight attribute accordingly.

4.2.2.5 Method 3: UpdateStages ()

The following details shall be defined for the methods:

- Purpose:
- Input: None.
- Output: updates the StageLeft or StageRight attributes accordingly.

4.2.2.5 Method 4: UpdateStatsOnUserFrames ()

The following details shall be defined for the methods:

- Purpose: updates the details such as CurrentAngleLeft, CurrentAngleRight, CountLeft, CountRight, stageLeft, stageRight on the application interface during the session along with the video stream.
- Input: None. (Uses class variables)
- Output: updates the application interface.

4.2.3 Class Name 3: ArmsReps

Basically, inherits ExerciseClass (base class for all exercises) and has attributes LeftArmCurlsCount and RightArmCurlsCount pertaining to Count attributes in the base class.

4.2.4 Class Name 4: Leg Curls

Basically, inherits ExerciseClass (base class for all exercises) and has attributes LeftLegCurlsCount and RightLegCurlsCount pertaining to Count attributes in the base class.

V. PROPOSED METHODOLOGY

In this chapter we discuss our method of solving the above-mentioned problem and the simple implementation pre-procedures required for the application proposed.

The method we have proposed involved constraint-based pose and stage estimation. These constraints define the pose of the user in frame and also decide the stage he/she is currently in, based on this info we can extract useful stats for the user. Furthermore, this does not involve any ML models for pose recognition, MediaPipe is the only tool, which is used to extract the skeleton of the user in frame, uses minimal ML and has low to nil delay. As mentioned in the previous section our project depends on this tool and assumes that it is close to accurate and reliable.

The main difference between the conventional ML model-based pose estimation and the constraint-based pose estimation is the ML model that is being built in the former approach, no such model building is required in our approach. Besides, in an ML model-based pose estimation approach it is infeasible to change the angles on the fly. Whereas in our approach the user can set his/her constraint angles and adjust the exercise according to their needs and comforts.

To date the basic approach for the workout application based on CV is done using some Deep Neural Network or some basic ML algorithms. As illustrated in Paper 1 and Paper 2 in our report, both these suffer from performance. To overcome the latency and improve performance, few frames were dropped, sometimes even useful user frames were lost. Even after this compromise, the resulting implementation was applicable only for a 2-stage exercise.

ML is an emerging topic. But that does not imply that it is applicable in every situation. Even though there is a slight improvement in accuracy this is compensated for by the performance of the system. To think alternatively, we can use simple constraints for each exercise, based on which the user frames are validated. This overcomes 2 issues:

1. Performance is improved as we are not using any models for comparison between user and reference frames.
2. New exercises can be built easily with only the dynamically set constraints.
3. The constraints, angles in this case, can be dynamically set in contrast to the previous approach.
4. The delay compromises for the lost accuracy
5. We can further improve system performance as we do not need to do affine transformations anymore.

International Journal of Engineering, Management and Humanities (IJEMH)
Volume 4, Issue 6, Nov.-Dec., 2023 pp: 106-119 www.ijemh.com

A simple example of such an implementation is shown,

Fig. 5a

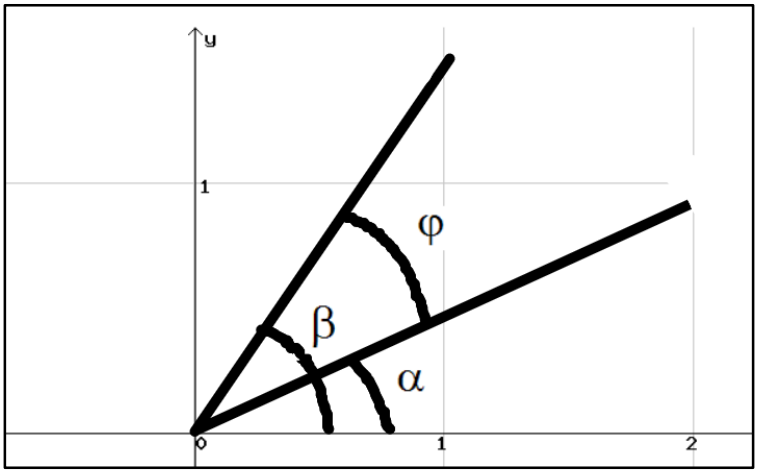

Fig. 5b

While doing arms curl it is necessary to fold arms to a particular angle and unfold to another angle. We calculate the angle using the following method: $\phi = \boldsymbol{\beta} - \boldsymbol{\alpha}$, where the point of intersection of the 2 lines is considered as the elbow. Where **α**, **β** and ϕ are the angles as shown in the above figures.

VI. IMPLEMENTATION AND PSEUDOCODE

6.1 Implementation Procedure

The application involves multiple modules such as front-end, back-end and the exercise module itself. Each of these were built in isolation and in different development cycles so integration of all the modules took a lot more time than expected. Considering all these and the ease of development of the exercise modules in python we decided to build the application completely in *Python.*

Previously we had planned to build the application as a web-based app using *Django*, but when the prototype was developed it failed to meet the performance expected. The frame rate dropped drastically from 60fps to 20-30fps. This was undesirable as the main goal of the constraint-based approach is to reduce the latency, delays and the frame drops.

This led to the idea of developing the application in a more application-like product than a web-based product. We prototype the application in multiple GUI libraries such as:

1. Python Kivy
2. Tkinter
3. PyQT5
4. WXPython

All these applications worked well in isolation but failed miserably when integrating with the back-end modules and the exercise modules. Few of them rejected camera access which was the bare minimum requirement for our application.

This is when we came across PySimpleGUI, which is both user-friendly and supports multiple platforms. This was created back in 2008, it is open source and supports cross platform development. It also allowed us to run the exercise modules, while itself running in the background. We could also include the graphs for the stats page without any difficulties.

The database used is SQLite which is compatible and convenient when working along with python. There are inbuilt modules in python to operate on the database. This database stores information related to each and every user, data such as name, emailID, age, etc. It also stores separate data for each user+ on the exercises performed by him/her.

6.2 The Exercise modules

As discussed, we have 2 exercises at present, arms curl and legs curl. There are many common python libraries for both the exercises modules these include:

1. OpenCV2
2. MediaPipe
3. Numpy

6.2.1 Arms Curl module

The arms curl module takes input as (left_max_angle, right_max_angle, left_min_angle, right_min_angle) and return the count of exercises performed by the user as follows: (counter_right,

counter_left, min_angle_left, min_angle_right, max_angle_left, max_angle_right). Each of these return variables are later updated in the database by the main function (calling function).
The pseudocode is as follows:
import necessary libraries
function definition with input parameter:
initialize all class variables -> default values
open loop for as long as the cam is open
read frames from the camera
make the frames writable
pass it through MediaPipe to obtain skeletal landmarks
extract required landmarks -> both side shoulder, both side elbow and both side wrist positions
calculate angle using the formula mentioned before
check if the angle falls within the constraint range
update count if required
update stage if required
update the visual feedback based on the current angle
Now write these data onto the frame so that it is visible to the user
Reps count, stage, current angle etc..
Resize the image if required
display the image
if for exit button triggering the termination of camera access thereby exiting the loop else loop again for the next frame
on exiting the loop consolidate the return values and return to the main function

6.2.2 Leg Curl Module
The arms curl module takes input as (left_max_angle, right_max_angle, left_min_angle, right_min_angle) and return the count of exercises performed by the user as follows : (counter_right, counter_left, min_angle_left, min_angle_right, max_angle_left, max_angle_right). Each of these return variables are later updated in the database by the main function (calling function).
The pseudocode is as follows:
import necessary libraries
function definition with input parameter:
initialize all class variables -> default values
open loop for as long as the cam is open
read frames from the camera
make the frames writable
pass it through MediaPipe to obtain skeletal landmarks
extract required landmarks -> both side ankle, both side knee and both side wrist hip
calculate angle using the formula mentioned before
check if the angle falls within the constraint range
update count if required
update stage if required
update the visual feedback based on the current angle
Now write these data onto the frame so that it is visible to the user
Reps count, stage, current angle etc..
Resize the image if required
display the image
if for exit button triggering the termination of camera access thereby exiting the loop else loop again for the next frame
on exiting the loop consolidate the return values and return to the main function

6.3 Database operations module
This module deals with all the operations pertaining to the database such as updates, fetch, delete, etc.. There are multiple functions defined in this module which will be discussed below:
1) dbHealthCheck: This module verifies if all the tables in the database are as per requirement and if any new table or rows need to be created. Pseudocode:
connect to the database/ create new one if the application is being opened for the 1st time
run query "**SELECT name FROM sqlite_master WHERE type='table';**"
check if "Profiles" table is present
if not create profile table using the query "**CREATE TABLE Profiles(USERNAME TEXT PRIMARY KEY NOT NULL,AGE INT,EMAIL varchar(200));**"
commit connections and return
2) dbVerificationFun : This function is used to authenticate the user, to check if the user data exists in the database. This function takes the username to be verified as input. Pseudocode:
connect to the database
execute query "**SELECT * FROM Profiles;** "
Check if the username exists in the table
if it is present return the user data
else return **None**
3) dbUpdateFun : This is used to perform update function of any kind, it requires the username, the value to be updated and the update value as input. Pseudocode:
connect to the database
check if there is a row for today
if present update " **UPDATE '{inp_uname}_stats' SET {update_key} = {update_key} + update_val WHERE Date = today_date;** "
else execute "**INSERT INTO '{inp_uname}_stats' VALUES (today_date, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)**;" and "**UPDATE '{inp_uname}_stats' SET {update_key} =**

International Journal of Engineering, Management and Humanities (IJEMH)
Volume 4, Issue 6, Nov.-Dec., 2023 pp: 106-119 www.ijemh.com

{update_key} + update_val WHERE Date = today_date;"
commit all and return

4) dbGetInfo : This function is used by the stats page to extract data for the current user and display the same in a visual format. This function takes in the username and the value to be extracted as input. Pseudocode:

connect to the database
extract the row pertaining to the username
check if it is armscurl or legscurl
    if armscurl return corresponding data
    else return legscurl related data
return

5) dbNewUserFun : This function is used when a new user is to be registered, this function handles all necessary operations to be performed when a new user needs to be added to the database. This takes in the username, age and emailID of the new user as the input. Pseudocode :

connect to the database
create necessary rows -> **con.execute("INSERT INTO Profiles (USERNAME, AGE, EMAIL) VALUES (?, ?, ?);",(inp_username, inp_age, inp_email))**
commit and return

6.4 The front-end modules

6.4.1 Login module

This module deals with the front-end for the login procedure. This has all the required input fields, once the user inputs the username and email id, this data is passed to the database module to authenticate the user. If the user is valid the login module moves to the next module. The login module also has access to the new user registration page. Pseudocode :

import necessary libraries and modules
display 2 input fields and 3 buttons
email id, username as input fields
login, exit and new user buttons for the respective actions
if event -> login
    dbVerificationfun is called, which returns if the user actually exists, and the credentials entered are correct
    if valid user :
        send the user to the disclaimer page
    else:
        show error
if event -> new user
    this calls the function/module that handles new user registration
else
    exit the application and save states

6.4.2 New User Registration module

This module deals with registering new users for the application. This module calls all the db functions that need to be called when registering a new user. It also does sanity checks on the new user's username and email ID. This module displays 3 input fields -> username, age and email ID. Pseudocode:

import all the necessary libraries and modules
define regex for email id
input fields display
once the user inputs the data run sanity check
if valid send to the database for creating the new user:
    if success, show message and return back to login page
else:
    display error

6.4.3 Main Page Module

This module has all the necessary controls for running all the features of the application like the exercise and stats. It has a drop down to select the exercise, 4 sliders to set the max and min angles for each side and exercise, 3 buttons -> for setting the angles, for starting the exercise and finally to show the stats. This module takes in the username as input, Pseudocode:

import all the available modules and necessary libraries
display all the input fields, slider and buttons
if event.start:
    if exercise is selected :
        run the respective exercise
        once the user returns from the exercise receive the return values and update the database
    else:
        show error messages
if event.set:
    set constraint angles as per slider value
if event.stats:
    send necessary values to the stats page module

6.4.4 The Stats page

This module deals with the statistical visualization of the progress of the user over time. It has 2 buttons, one for each exercise type, in each exercise there are 2 graphs, one for the count and the other for the max and min angle.import necessary modules and libraries like matplotlib for the graph
use the username to get the user's data from the database

if a graph already exists, destroy it draw new graph for the data of the user
return to the main page when exit is pressed

VII. RESULTS AND DISCUSSION

The application we built now works well with almost all types of systems. The need for heavy processing tools has been removed, this application now also works for multiple users. Unlike the ML based projects, we are able to give real time results for all the exercises performed by the user. The application is now able to process user frames in sync with the input frames i.e., processing 60 frames per second.

Since we have scrapped the idea of storing the frames and analysing it in the future or with some delay and store only relevant and useful data so that it helps the user improve his routine exercise, we are saving a lot more storage space and the application comes down to a few MBs as compared to ML based stored analysis which involves excessive use of storage space costing few GBs.

We can clearly see the improvement in the performance and efficiency of the application. The ability of dynamically setting the angles in the application gives the user an unique experience where they are able to control their own exercise in a way they want without building a new ML model just to fix their angles.

The application now also has a statistics page where the user can see their daily activities and visualize their improvement over time. This helps them set their angles for future exercise sessions. The on-screen visual feedback helps them fix their poses during the exercises and the angle and counter being displayed alongside the user give them a bio-feedback mechanism.

VIII. CONCLUSION AND FUTURE WORK

As discussed before the application is dependent on the MediaPipe module for providing the skeletal structure of the user in frame, we assume here that the module provides the output in bare minimum time and with good accuracy. Technically the speed of the application depends directly on this module (being the slowest of all the other operations and modules).

We have also discussed that this application is solely dependent on constraints such as angles, inclination, etc,., so complex poses need more constraints. In some cases when the poses are too complex for constraints to be built ML models fit well in these situations. Side-faced user actions are not as accurate as the front-faced exercises but performs better than the ML model in this aspect.

To improve this project, we can go ahead with building more constraints or build constraints for new exercises. Side-faced detection can be improved using 2 cameras, where the user's action can be modeled in a 3D domain. Stereo vision gives a better understanding of the posture and further improves the accuracy.

Further audio feedback to correct the exercise can be used to make it an immersive experience for the users. Through augmentation we can simulate the actual exercise so that users can compare themselves with a reference video. One more step can be taken to calibrate and correct the user's spine/backbone correction through the angles(inclination) detection, this needs the user to be facing sideways.

REFERENCES

[1]. A. Nagarkoti, R. Teotia, A. K. Mahale and P. K. Das, "Realtime Indoor Workout Analysis Using Machine Learning & Computer Vision," 2019.

[2]. Nitesh Sonwani, Aryan Pegwar, “Auto_Fit: WORKOUT TRACKING USING POSE-ESTIMATION AND DNN”, International Journal of Engineering Applied Sciences and Technology, 2020 Vol. 5

[3]. Vivek Veeriah J. and Swaminathan P. L, “Robust Hand Gesture Recognition Algorithm for Simple Mouse Control”, International Journal of Computer and Communication Engineering, 2 March, 2013.

[4]. Bidyut Jyoti Boruah, Anjan Kumar Talukdar and Kandarpa Kumar Sarma, “Development of a Learning-aid tool using hand gesture based human Computer Interaction System”, 2021 Advanced Communication Technologies.

[5]. Roshnee Matlani, Roshan Dadlani, Sharv Dumbre, Shruti Mishra and Abha Tewari “Virtual Mouse using Hand gesture” Year of publication: 2021.

APPENDIX

1. OpenCV2 : Is an open source module for image processing in python, it also allows for easy camera access

2. MediaPipe : Open source tool for human body mapping

3. Matplotlib : Tool for plotting graphs of many kinds, extensively used for data analysis in python

4. **PySimpleGUI :** It is a simple open source tool for building a front-end for the users which is both convenient and efficient.
5. **Numpy :** Is a python library used to work with huge multi dimensional arrays
6. **SQLite :** Is an open source database engine which interfaces well with python. Use text based storage for tables.
7. **KIVY :** Is an open source tool for building multi touch GUI based software application and mobile apps
8. **PyQT5 :** python cross platform GUI toolkit