TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

ANALYSIS OF POSTURE IN GYM BASED EXERCISES USING HUMAN POSE ESTIMATION

Prepared By:

Rabindra Regmi 073BCT531
Sabal K.C. 073BCT537
Subash Tiwari 073BCT544
Sujeet Silwal 073BCT546

Submitted To:
Department of electronics and computer engineering
lalitpur,nepal

March,2020

Acknowledgement

We would like to express our sincere gratitude to the **Department of Electronics and Computer Engineering,** Pulchowk Campus, Institute of Engineering for providing us with this wonderful opportuntity and the great platform to apply and extend our knowledge gained in past years performing a real world project. Our special thanks goes to our supervisor **Bibha Sthapit** whose contribution in stimulating suggestions and encouragement helped us a lot for initiating this project.

We would like to express our deep gratitude to **Mr. Sumit Shrestha**, instructor of New Perfect Gym, Nayabazar, Kathmandu for providing us the details rule about the exercises so we could integrate those rules into our project.

Our special thanks goes to all the teachers who taught us in the past three years and made us capable to do this project. We would also like to express our gratitude to our classmates for giving feedback on our ideas.

List of Figures

Contents

1 Introduction

1.1 Background

Human pose estimation refers to the process of inferring poses in an image. Essentially, it entails predicting the positions of a person's joints in an image or video. This problem is also sometimes referred to as the localization of human joints. It's also important to note that pose estimation has various sub-tasks such as single pose estimation, estimating poses in an image with many people, estimating poses in crowded places, and estimating poses in videos. It can be used in various different fields but our goal is to use this data of joints from pose estimation to analyze different exercises.

Exercise is very beneficial to the health if done correctly but also can be very dangerous if posture and form is not correct. Some exercises such as deadlift can cause big accident if form is not correct. Our idea for this project is to develop a software or mobile app which can identify whether the person is performing exercise correctly or not and give some feedback on how to correct form. There's always more number of people in the gym with respect to the number of trainers, so the trainer can't keep an eye on everyone. That might cause problems for those persons who are out of reach of a trainer at that moment. Our software,SmartTrainer, will try to minimize that problem by monitoring those people and giving them proper feedback. This will help to reduce the dependency on trainers and also provide an alternate overview while performing exercises.

First part of SmartTrainer is Human Pose Estimation, which is very difficult but highly applicable field in Computer Vision. Pose estimation is critical for problems involving human detection and activity recognition, and can also aid in solving complex problems involving human movement and posture.

Second Part of SmartTrainer is to monitor individual person and giving proper feedback on the exercise they are doing. For this, we are planning to use rule based system using poses and instruction from personal trainers and other qualified persons as the ground truth for proper form. Our plan is to have a complete software which consists of two concepts that were previously mentioned with feature of recording video to monitor and keep track of user's exercise routine. It will also have feature of counting numbers of correct reps and sets and will also record user progress on exercise.

1.2 Objectives

This project's main aim is single pose estimation of human from video and correcting posture of person while doing exercise and giving proper feedback to the users.

The main objectives of this project can be listed as:

- Estimate key joint of human body using Pose Estimation
- Almost real time Pose Estimation
- Detecting quality of exercise person is doing
- Provide proper feedback to the user (where the person can improve or any incorrectness)
- Integrate proper interface to show all exercises and feedbacks

1.3 Scope

The project mainly focuses on pose estimation of human body. Pose Estimation has many applications like, for high-accuracy human detection for robots and intelligent driver assisting system like autonomous cars for detection of pedestrians, for character animation of humans, for video games, for medicals applications like detecting postural issues such as scoliosis based upon posture.

Our project focuses on posture correction in gym based exercise by monitoring and giving feedback. We take points from pose estimation and analyse it using different technique and implement rule to know if the exercise performed was correct or not.

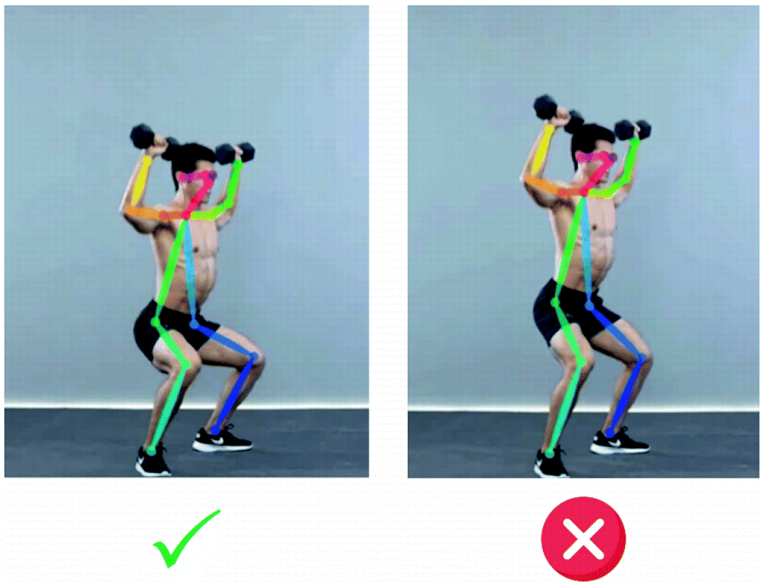


Figure 1: Training Analysis

2 Literature Review

In our initial research, we found out that numerous research works have been conducted on the field of human pose estimation and few in the field of correcting posture in gym based exercise. Also, many researchers from all around the globe have used various ways to achieve their goals. Alexander Toshev and Christian Szegedy used Deep Neural Network method for human pose estimation[1]. The pose estimation was formulated as a DNN-based regression problem towards body joints.

Also, Novel architecture for Human pose estimation has also been proposed by researchers from New York University. This includes an efficient 'position refinement' model that is trained to estimate the joint offset location within a small region of the image. This refinement model is jointly trained in cascade with a state-of-the-art ConvNet model to achieve improved accuracy in human joint location estimation.[2].

Human Pose Estimation using Iterative error feedback method[3] has also been proposed. Here they proposed a framework that expands the expressive power of hierarchical feature extractors to encompass both input and output spaces, by introducing top-down feedback.

Similar research has been done by using convolutional pose machines [4].Pose Machines provide a sequential prediction framework for learning rich implicit spatial models. In this work they showed a systematic design for how convolutional networks can be incorporated into the pose machine framework for learning image features and image-dependent spatial models for the task of pose estimation.

Similarly, there have been many researches in the field of exercise posture correction. Recently, Richard Yang and Steven Chan in 2018 proposed a PoseTrainer Model to correct exercise posture using PoseNet. They used Heuristic Based system to detect quality of exercise.

Openpose:

It uses part affinity fields. It provides a real-time method for Multi-Person 2D Pose Estimation based on its bottom-up approach instead of detection-based approach in other works. First, the image is passed through a baseline network to extract feature maps. Then, the feature maps are processed with multiple stages CNN to generate: 1) a set of Part Confidence Maps and 2) a set of Part Affinity Fields (PAFs). Part confidence maps are a set of 2D confidence maps S for body part locations. Each joint location has a map. Part Affinity Fields (PAFs) are a set of 2D vector fields L which encodes the degree of association between parts. Finally, the Confidence Maps and Part Affinity Fields are processed by a greedy algorithm to obtain the poses for each person in the image.

Posenet:

They have used an efficient 23 layer deep convnet, demonstrating that convnets can be used to solve complicated out of image plane regression problems. This was made possible by leveraging transfer learning from large scale classification data.

Application in health/exercise

Similarly, there has been much research in the field of exercise posture correction. Recently, Richard Yang and Steven Chan[5] in 2018 proposed a PoseTrainer Model to correct exercise posture using PoseNet. They have used a Heuristic Based system to detect quality of exercise. They collected data for several exercises such as bicep curl, front raise, shoulder shrug and shoulder press which includes dataset for correctly performed exercise and incorrectly performed exercise. Firstly, they use openpose to get the pose sequence data of the exercise. (x,y) coordinate and confidence of each of the 18 major joints: left ear, right ear, left eye, right eye, nose, neck, left shoulder, left elbow, left wrist, right shoulder, right elbow, right wrist, left hip, left knee, left

ankle, right hip, right knee and right ankle is the output of openpose. Normalization is carried out to account for difference in measurements. Since torso length is constant, it is used as the reference for normalization. Then, perspective detection is used to find which side is being used - right or left. After this, for each exercise, heuristics are used. For example, angle between upper arm vector and torso vector is used in bicep curl analysis. They have also used machine learning based approach, namely DTW (Dynamic Time Warping). Since we have a temporal sequence and time taken for exercise is not uniform, the authors have proposed DTW distance to calculate the difference in the correct exercise and user performed exercise.

3 Methodology

3.1 System Block Diagram

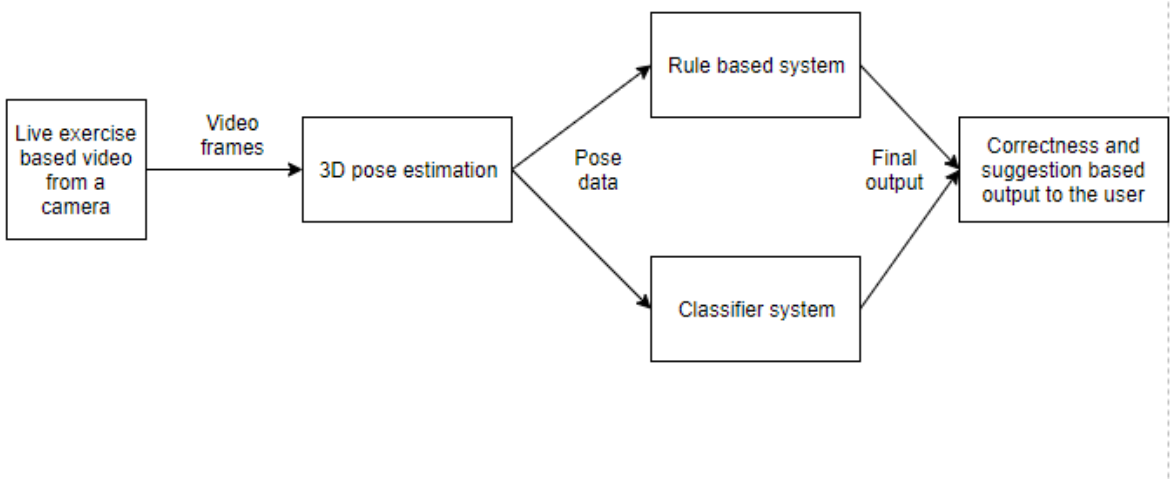


Figure 2: Block Diagram of System

Our system will consist of a camera which may be of any type, for example, a mobile phone camera or a webcam from a laptop computer. The camera will be used to capture a normal RGB format video whose frames are fed into the 3D pose estimation model. The video will be such that the person performing an exercise routine is clearly visible. Pose estimation is performed to get the detailed pose data and this data is then fed to a rule based system as well as a classification and comparison based system to analyze the exercise routine. If the exercise is performed correctly, the repetition count of the exercise is automatically recorded and displayed in the user interface of our application. If the person does not perform the exercises according to the rules, then feedback is generated to notify the user and display the appropriate steps for correction.

3.2 General Steps Followed

Our project will consist of two major components - pose estimation and analysis of gym/exercise posture.

The pose estimation part can be described as: Given a video of a person who is performing a given exercise, a 3d pose estimation model is built with the task of producing a 3D pose that matches the spatial position of the depicted person. The following steps are followed:

3.2.1 Data Preprocessing and feature extraction

Firstly, the image/video data has to be preprocessed before being fed into the model. It includes the following steps

Background removal: It is used to segment the background and remove any noise in the image. Foreground detection is one of the major tasks in the field of computer vision and image processing whose aim is to detect changes in image sequences. Background subtraction is any technique which allows an image's foreground to be extracted for further processing (object recognition etc.).

Bounding box creation: In digital image processing, the bounding box is merely the coordinates of the rectangular border that fully encloses a digital image when it is placed over a page, a canvas, a screen or other similar bi-dimensional background. Some algorithms, especially in MPPE, create bounding boxes for every human present in the image. Bounding box helps to separate the required human subject from the rest of the surroundings. This is also used in many algorithms of pose detection.

Camera calibration and image registration: Image registration is required in case inputs from multiple cameras are used. In case of 3D Human Pose Estimation, camera calibration also helps in converting the reported groundtruth into standard world coordinates.

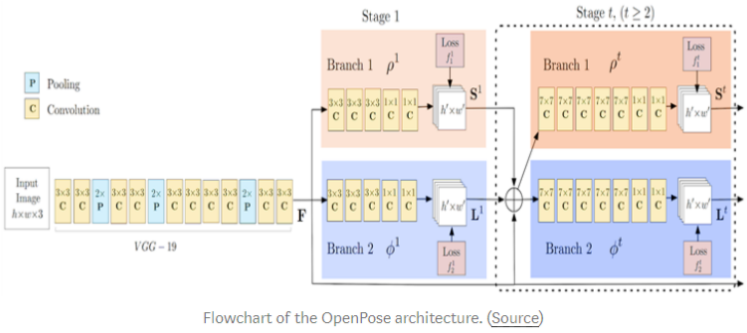
Feature extraction is another important part of machine learning which refers to the creation of derived values from raw data (such as an image or video in our case), that can be used as input to a learning algorithm. Features can either be explicit or implicit. Explicit features include conventional Computer Vision based features like Histogram of Oriented Gradients (HoG) and Scale Invariant Feature Transform (SIFT). These features are calculated explicitly before feeding the input to the following learning algorithm. Implicit features refers to deep learning based feature maps like outputs from complex Deep Convolutional Neural Networks (CNNs). These feature maps are never created explicitly, but are a part of a complete pipeline trained end-to-end.

3.2.2 Pose estimation model

One of the ways in predicting joint locations is producing confidence maps for every joint. Confidence maps are probability distribution over the image, representing the confidence of the joint location at every pixel. There are many approaches which can be used for the pose estimation tasks. Most of the popular methods involve deep learning techniques. Some of the methods that can be used are as follows:

OpenPose: OpenPose is one of the most popular bottom-up approaches for multi-person human pose estimation. As with many bottom-up approaches, OpenPose first detects parts (key-point) belonging to every person in the image, followed by assigning parts to distinct individuals. The figure shows the architecture of the model:

The OpenPose network first extracts features from an image using the first few layers (VGG-19 in the above flowchart). The features are then fed into two parallel branches of convolutional layers. The first branch predicts a set of 18 confidence maps, with each map representing a particular part of the human pose skeleton. The second branch predicts a set of 38 Part Affinity Fields (PAFs) which represents the degree of association between parts. Successive stages are used to refine the predictions made by each branch. Using the part confidence maps, bipartite graphs are formed between pairs of parts (as shown in the above image). Using the PAF values, weaker links in bipartite graphs are pruned. Through the above steps, human pose skeletons can be estimated and assigned to every person in the image. The method can be shown in figure as follows:



Flowchart of the OpenPose architecture. (Source)

Figure 3: Open Pose Architecture

Vnect: CNN Pose Regression: The core of our method is a CNN that predicts both 2D, and root (pelvis) relative 3D joint positions in real-time. The new proposed fully-convolutional pose formulation leads to results on par with the state-of-the-art offline methods in 3D joint position accuracy . Being fully-convolutional, it can operate in the absence of tight crops around the subject. The CNN is capable of predicting joint positions for a diverse class of activities regardless of the scene settings, providing a strong basis for further pose refinement to produce temporally consistent full-3D pose parameters.

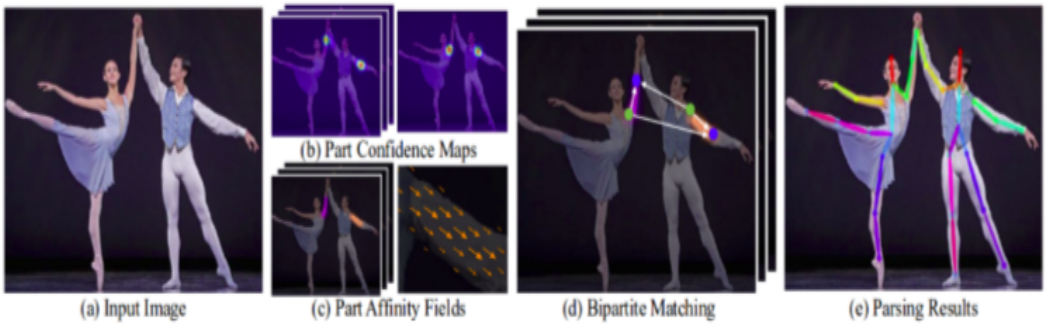


Figure 4: Vnect

Kinematic Skeleton Fitting: The 3D predictions from the CNN along with the previous frames can used to generate a consistent full 3D skeletal pose. Smoothing and filtering can be done to generate a smooth transition between the frame. The obtained armature from the output of the CNN can be mapped to the bounding hierarchy. The position of the different armature obtained from the CNN and are made relative to each other. This process later simplify setting up the animation system. Also the size of the mesh that we want to animate may vary in size, this can also be accommodated by tracking the pose in true metric space and then scaling them to fit the required mesh. The scaling can done by generating a simple bounding box or by using advanced body part estimation technique for more accurate result.

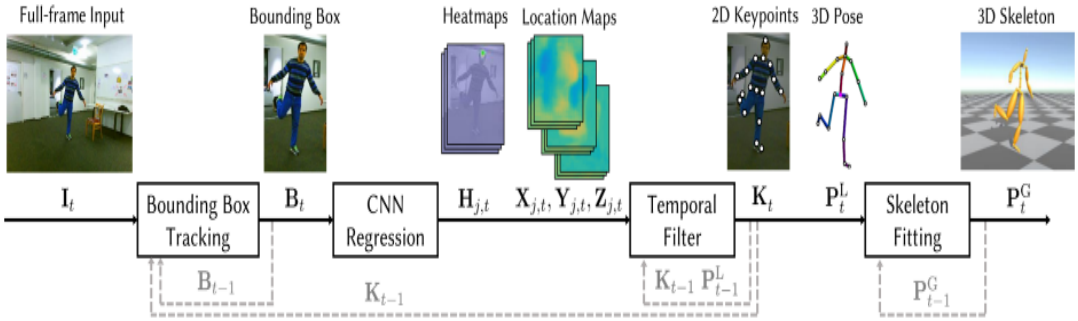


Figure 5: Kinematic Skeleton Mapping

Rule based system for exercise evaluation: For evaluating exercise posture, firstly, the output of pose estimation will have to be parsed into proper format. The full skeletal estimation of the person is obtained and necessary normalization is performed. We can use length of torso as the standard for normalization. The perspective of the user is also determined.

The rule based system will be according to the standard guidelines of a particular exercise.

Geometry based heuristics and thresholds can be used to determine whether the given exercise has been performed correctly or not. Along with that, the actual mistake performed by the user can also be pointed out. For instance, in a bicep curl, we can identify two heuristics of interest. First, the upper arm should be kept steady and not move significantly. This can be quantified by the angle between the upper arm vector and the torso vector. If the upper arm is held steady, then it should be parallel to the torso with minor variations for the entire video. Second, a proper, complete curl requires the weight to be brought up until the bicep is fully contracted, beyond the midway point (90between upper arm and forearm) that is commonly stopped at. This improper form is typically a result of the user using weights that are too heavy. We can quantify this problem by the minimum angle achieved between the upper arm and forearm. In the start position with the weight down, the angle should be nearly 180. As the weight is lifted, the angle should decrease until when the user stops, and increase again as the weight is brought down. Such rules can be used for many other exercises.

3.2.3 Machine Learning based evaluation

Another approach to evaluate the exercise posture can be through various machine learning methods.

Dynamic Time Warping (DTW): Since the recorded videos can be of arbitrary length, this results in a different keypoint vector length for each example. Different feature vector lengths present a challenge for many machine learning models, so dynamic time warping (DTW) can be used. DTW is a metric used to measure the non-linear similarity between two time series. When two similar sequences are phase-shifted (i.e. shifted in the time dimension), metrics like the Euclidean distance fail because it is a direct comparison of two points at the same time. In DTW, we can dynamically identify the keypoint in the second sequence that corresponds to a given point in the first. Hence, DTW distances can be calculated and nearest neighbor classification can be performed to produce the output of correct or incorrect for a given exercise sequence.

Recurrent Neural Networks(RNN) and Long Short Term Memory (LSTM): Recurrent Networks can remember the past, and are thus preferred for sequence processing. RNN cells are the backbone of Recurrent Networks. RNN cells have 2 incoming connections, the input and the previous state. Similarly, they also have 2 outgoing connections, the output and the current state. This state helps them combine information from the past and the current input. A simple RNN cell is too simplistic to be uniformly used for time series analysis across various domains. So multitudes of variations have been proposed over the years to adapt Recurrent Networks to the various domains. They include LSTMs and GRUs. These can be used in such time series based problems. The problem with using this sort of approach is that a large amount of data may be required for higher accuracy.

Convolutional Neural Networks (CNN) - LSTMs: Since the state information travels through every timestep, RNNs are capable of remembering only the recent past. Gated networks like LSTMs and GRUs on the other hand can handle comparatively longer sequences, but even these networks have their limits. For a better understanding of this issue, we can look into vanishing and exploding gradients which happens for very deep networks. So, for long sequences we have to shorten the sequence. One way is to discard the fine grained time information present in the signal. This can be done by accumulating small groups of data points together and creating features from them, which are then passed, acting like a single datapoint, to the LSTM. This process is done by CNN and the data points can be passed to the recurrent units.

3.3 Available datasets for pose estimation

Available datasets for Pose Estimation are

MPII: The MPII human pose dataset is a multi-person 2D Pose Estimation dataset comprising of nearly 500 different human activities, collected from Youtube videos. MPII was the first dataset to contain such a diverse range of poses and the first dataset to launch a 2D Pose estimation challenge in 2014.

COCO : The COCO keypoints dataset is a multi-person 2D Pose Estimation dataset with images collected from Flickr. COCO is the largest 2D Pose Estimation dataset, to date, and is considering a benchmark for testing 2D Pose Estimation algorithms.

HumanEva : HumanEva is a single-person 3D Pose Estimation dataset, containing video sequences recorded using multiple RGB and grayscale cameras. Ground truth 3D poses are captured using marker-based motion capture (mocap) cameras. HumanEva was the first 3D Pose Estimation dataset of substantial size.

Human3.6M : Human3.6M is a single-person 2D/3D Pose Estimation dataset, containing video sequences in which 11 actors are performing 15 different possible activities were recorded using RGB and time-of-flight (depth) cameras. 3D poses are obtained using 10 mocap cameras. Human3.6M is the biggest real 3D Pose Estimation dataset, to date.

3.4 System Implementation

Tensorflow : TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.

Tensorflow Lite : This will be used if our project will include mobile device application. TensorFlow Lite is an open source deep learning framework for on-device inference. It can be used to deploy machine learning models in mobile and IOT devices.

OpenCV : OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

3.5 Final System Testing and Documentation

3.5.1 Test-driven deployment

Test-driven development (TDD) is a software development process that relies on the repetition of a very short development cycle: requirements are turned into very specific test cases, then the software is improved so that the tests pass. This is opposed to software development that allows software to be added that is not proven to meet requirements.

3.5.2 Agile software development

Agile software development comprises various approaches to software development under which requirements and solutions evolve through the collaborative effort of self-organizing and cross-functional teams and their customers. It advocates adaptive planning, evolutionary development, early delivery, and continual improvement, and it encourages rapid and flexible response to change.

4 Theoretical Background

4.1 Number of people being tracked and use of image/video in pose detection:

One of the parameters in pose estimation is the number of people which are being tracked. By this parameter, pose estimation can be classified into Single-person and Multi-person pose estimation. Single-person pose estimation (SPPE) is the method in which only one person is present in the frame. On the other hand, Multi-person pose estimation (MPPE) needs to handle the additional problem of inter-person occlusion. Occlusion is the condition which occurs if an object you are tracking is hidden (occluded) by another object. Initial approaches in pose estimation were mostly focused on SPPE, however with the availability of huge multi-person datasets, the MPPE problem has lately been getting increased attention. Either image or video can be used in the task of pose detection. A video is nothing but a collection of images, where every two consecutive frames share a huge portion of the information present in them (which is the basis of most of the video compression techniques). These temporal (time based) dependence in videos can be exploited while performing Pose estimation. For a video, a series of poses need to be produced for the input video sequence. It is expected that the estimated poses should ideally be consistent across successive frames of video, and the algorithm needs to be computationally efficient to handle large number of frames. The problem of occlusion might be easier to solve for a video due to the availability of past or future frames where the body part is not occluded. If temporal features are not a part of the pipeline, it is possible to apply static pose estimation for each frame in a video. However, the results are generally not as good as desired due to jitters and inconsistency problems. Hence, image or video representations can be used. For animation purposes, video is preferred as it can provide real time and continuous representation of a movement to be represented.

4.2 Input Modality:

Modality refers to the different types of inputs available. The input video/image may have different parameters. The modes of input can be as follows:

4.2.1 RGB image/video:

The images that we see around us on a daily basis, and the most common type of input for Pose Estimation. Models working on RGB-only input have a huge advantage over others in terms of the mobility of the input source. This is due to the ease of availability of common cameras (which capture RGB images), making them the models that can be used across a huge number of devices.

4.2.2 Depth information included image/video:

In a Depth image, the value of a pixels relates to the distance from the camera as measured by time-of-flight. The introduction and popularity of low-cost devices like Microsoft Kinect has made it easier to obtain Depth data. Depth image can complement RGB image to create more complex and accurate Computer Vision models, whereas Depth-only models are vastly used where privacy is a concern.

4.2.3 Infra-red (IR) image/video:

In an IR image, the value of a pixel is determined by the amount of infrared light reflected back to the camera. Experimentation in Computer Vision based on IR images are minimal, as compared to RGB and Depth images. Microsoft Kinect also provides IR image while recording. However, currently there are no datasets that contain IR images. IR based systems can have several other disadvantages as well.

4.3 2D vs 3D Pose Estimation:

The output of pose estimation can be expressed in either two or three dimensions. Depending on the output dimension requirement, the Pose Estimation problem can be classified into 2D Pose Estimation and 3D Pose Estimation. 2D Pose Estimation is predicting the location of body joints i in the image (in terms of pixel values). On the other hand, 3D Pose Estimation is predicting a three-dimensional spatial arrangement of all the body joints as its final output. It consists of the prediction of the z coordinate as well. Most 3D Pose Estimation models first predict 2D Pose, and then try to lift it to 3D Pose. However, some end-to-end 3D Pose Estimation techniques also exist which directly predict 3D Pose.

5 Task Completed

5.1 Data Collection:

We obtained data from various locations like youtube and gym consulting websites. We also recorded some exercise at home performing by ourselves. Data then was refined and analysed.

5.2 Pose Estimation:

We tried different library which would give us the result as we expect. We tried various pose estimation library like posenet and tf-pose- estimation, but we found out openpose suits our application the best so, we used openpose. We send the video to open pose which we require to get pose of and use that points to analyse the output of how user is performing the exercise.

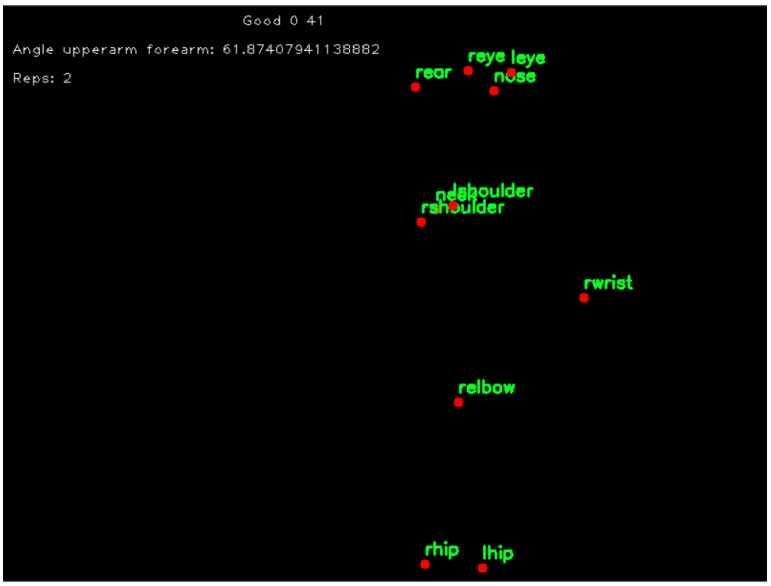


Figure 6: Pose Estimation Plot

In above figure, we have plotted all the joints obtained from poseestimation and plot was plotted and line was drawn between the moving joints and angle was visualized.

5.3 Data Visualization

Multiple videos of good exercise and bad exercises were sent to openpose and output joints was saved as numpy arrays and was plotted in opencv to visualize between datas. It is then used to make rules of exercises, what exercise is good and what is bad.

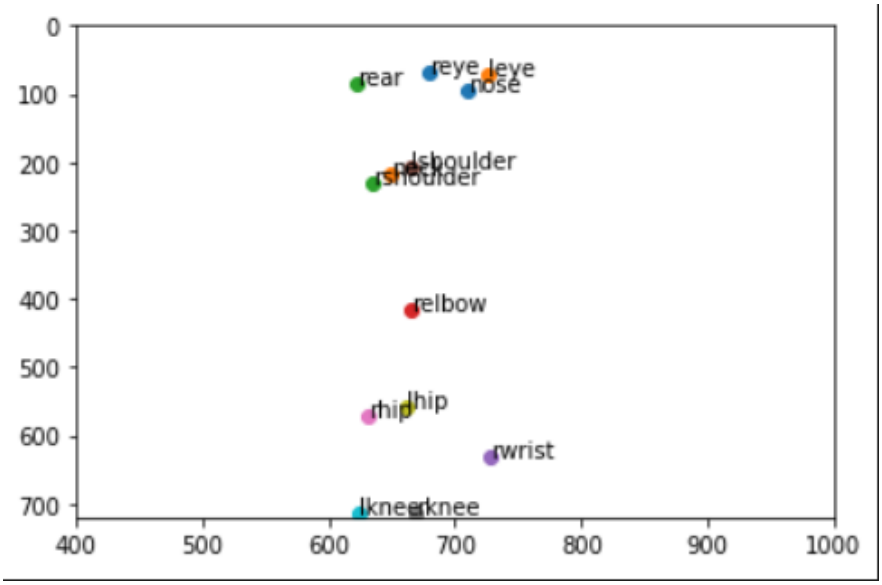


Figure 7: Visualization Plot 1

```
Data shape:  (114, 18, 3)
Mean torso:  360.11485467535044
114
nose <1.9708212276881114, 0.26258344739832407, 0.835979>
neck <1.8021833633746602, 0.6053263206721059, 0.584495>
rshoulder <1.763887248063247, 0.6434558224733541, 0.613341>
relbow <1.845452891964497, 1.1546371792267567, 0.54861>
rwrist <2.0198889064316887, 1.7531906607106573, 0.78617>
lshoulder <1.845719473580761, 0.5780572428417762, 0.333104>
lelbow <0.0, 0.0, 0.0>
lwrist <0.0, 0.0, 0.0>
rhip <1.7532267603045266, 1.5844972585604844, 0.261292>
rknee <1.8565076983639413, 1.987218773980178, 0.137317>
rankle <0.0, 0.0, 0.0>
lhip <1.8347313125853828, 1.551885440837532, 0.124535>
lknee <1.731453151417804, 1.9871632361434566, 0.101476>
lankle <0.0, 0.0, 0.0>
reye <1.8891583925725985, 0.19182796572575952, 0.935494>
leye <2.0141213020881454, 0.19704741162086012, 0.914553>
rear <1.7261881644965915, 0.235362132107575, 0.886749>
lear <0.0, 0.0, 0.0>
```

Figure 8: Visualization Output

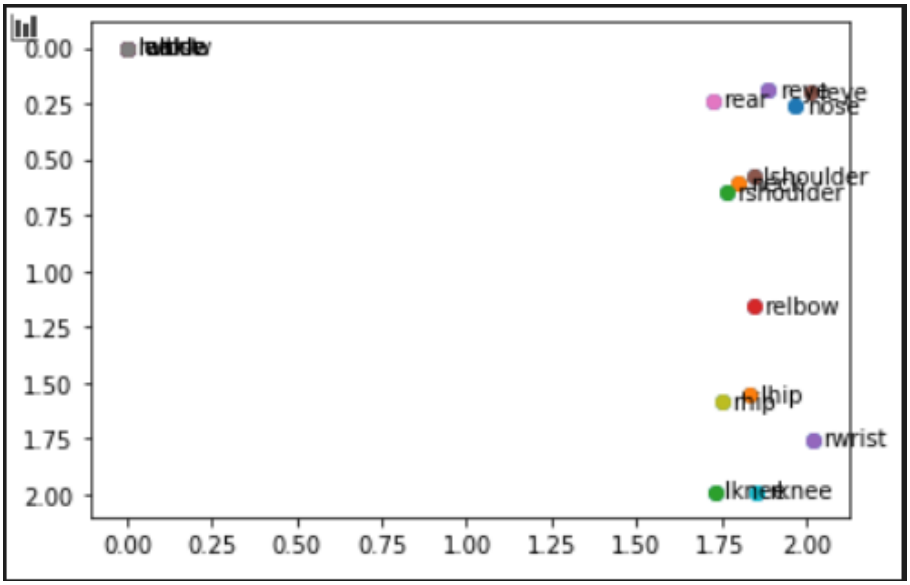


Figure 9: Visualization Plot 2

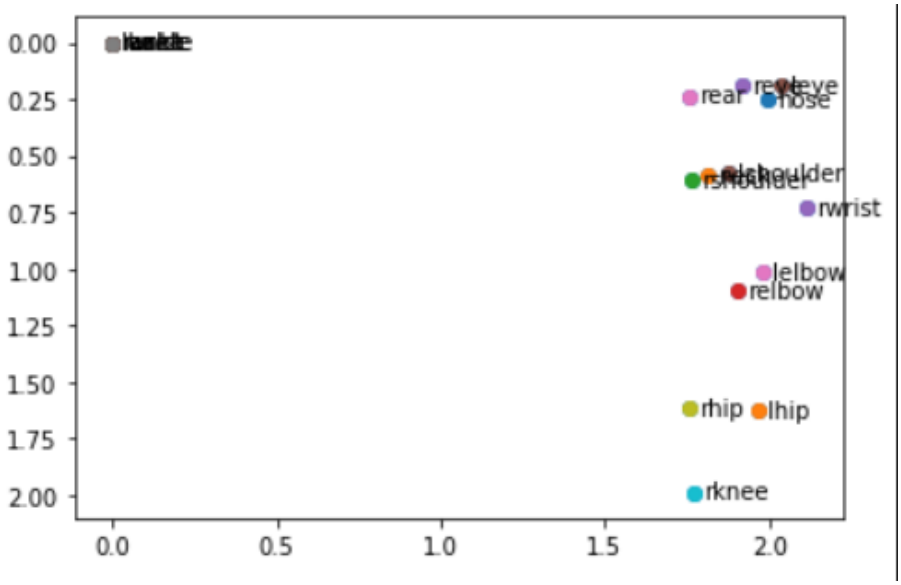


Figure 10: Visualization Plot 3

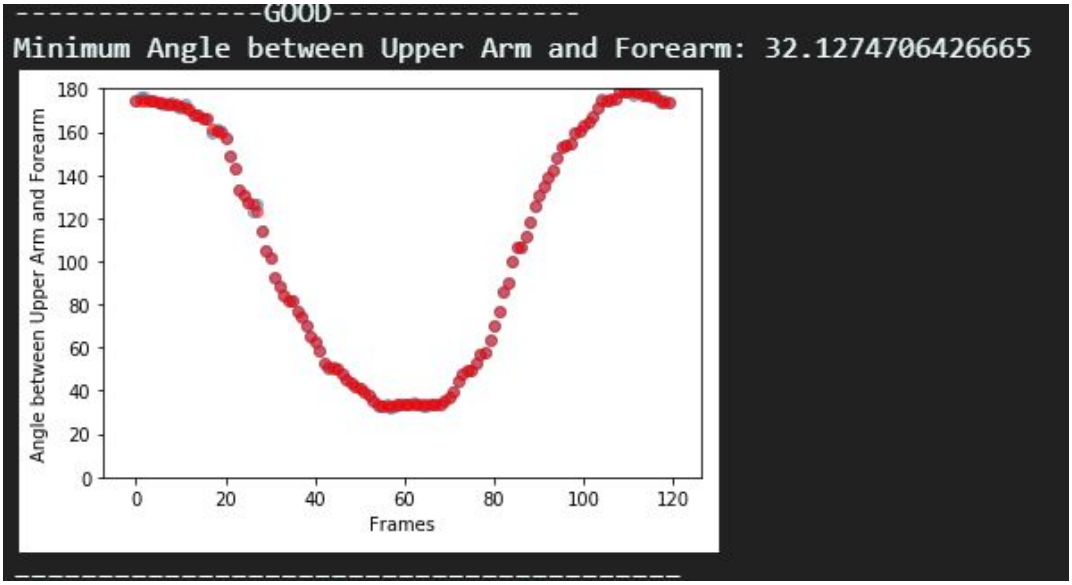


Figure 11: Good exercise I

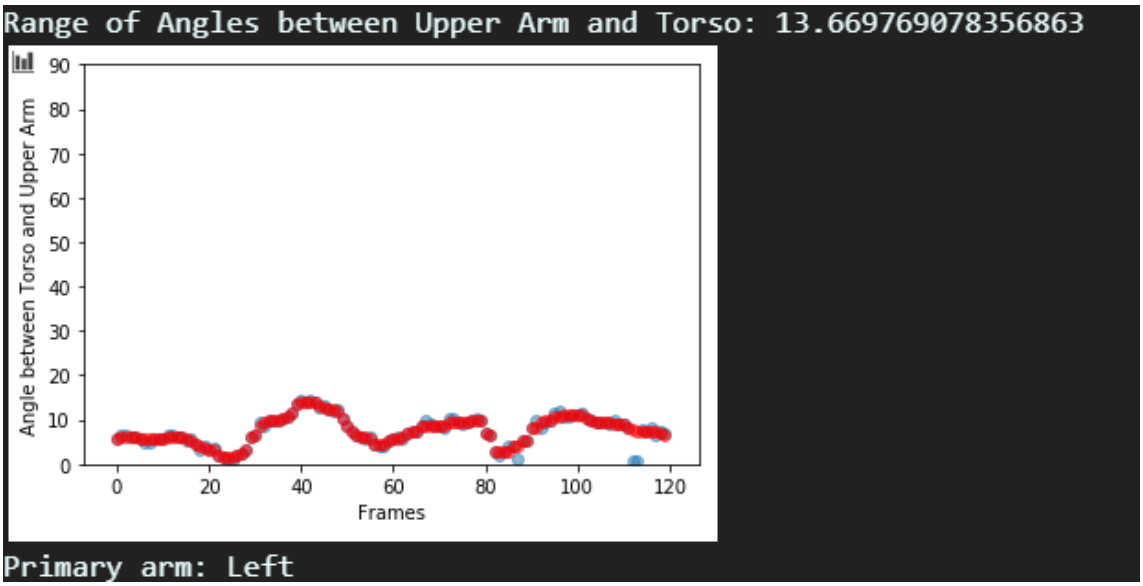


Figure 12: Good exercise II

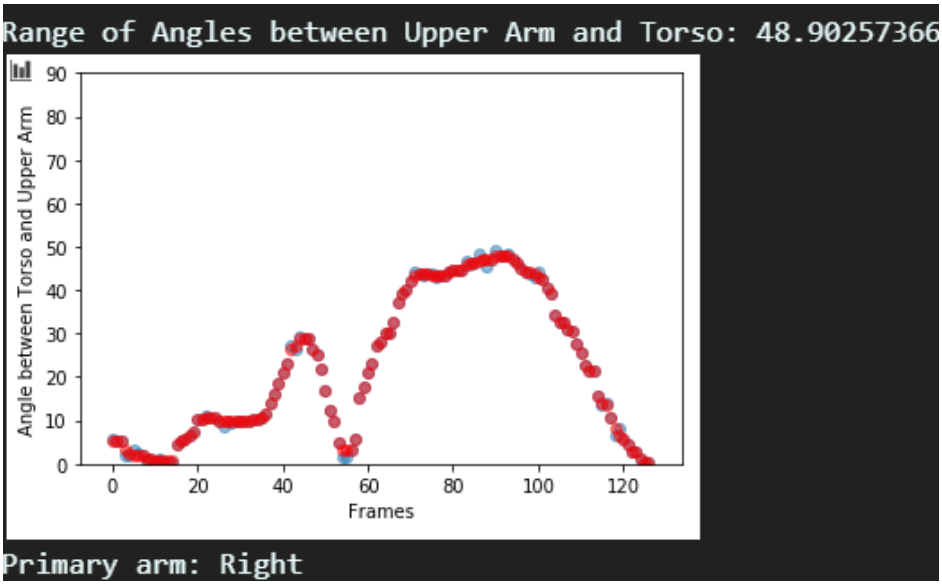


Figure 13: Bad exercise I

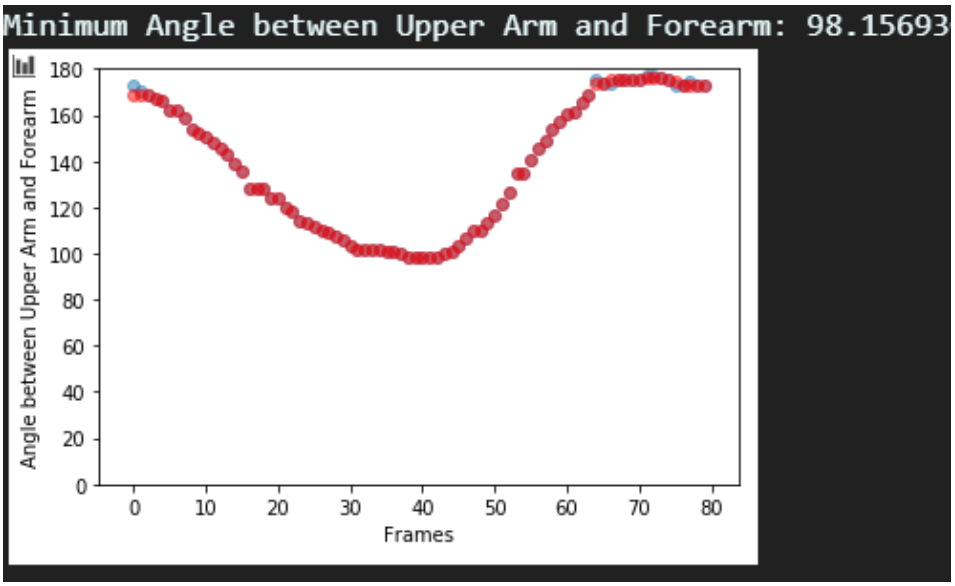


Figure 14: Bad exercise II

5.4 Implementing rules

After visualizing the data, we then went to gym trainer to ask how can we analyse rules to make it onto our project. We listed all the rules which we could follow to our project, calculated the mathematical form of the words used by the trainer and implemented the rule which is correct one and which one is not.

Bicep Curl Rules: (From Trainer)

- Your shoulder-elbow part should not move.
- Your body should not move backwards.
- While going up, your bicep should be fully squeezed and while going down your hand should be straight and bicep should fully expanded.

- Your wrist should not move.
- You have to complete full range of motion for better efficiency.
- You have to release weight with full control and slowly while going down.

Bicep Curl: (From our perspective):

- Torso Vector and Shoulder-Elbow vector should be parallel
- Neck-Mid-Hip/Torso vector should not move.
- Forearm vector (elbow-wrist) motion should be full range.
- Angle between forearm vector and shoulder vector should me minimum while going up and should be 180-degree to complete one set.

6 Task Remaining

6.1 Interface

Now, we have seen the results only in openCV and we have specified if exercise is right or wrong in openCV window. But we need to implement the interfaces mobile app and web app to show everything of users. Users could see their improvements and in which exercise they should focus on. The current interface is only a type of debugging window where we have visualized the output and tested rules of our system.

6.2 Giving feedback with animation or video

If user performs the exercise in a wrong way continuously then we generate the animations showing the position should be corrected to look like position in animations. This may help user to improve. The ground truth animation can be shown as a visual hint to the user.

6.3 Nearly Real Time

Results obtained shown so far are not real time. First we get joints first by sending entire video and analyse it but our results should be near real time. The live video feed must be sent to the backend where live inference is done and the output is sent back to the user.

7 Problem Faced So Far

7.1 Data Collection:

Data we collected was very raw. 10mins or more video had only 1 or 2 min of useful information also in different parts. So it was difficult to r refine data we had got.

7.2 Computational Problem:

To perform real time, we need very high specifications hardware, which is to expensive so there is problem taking pose estimation in real time. While performing pose estimation in real time, it gives fps of 3 to 5. To improve it better hardware should be used.

8 Expected Outcome

Figure 15: Expected Outcome

Figure 16: Expected Outcome

We seek to output the result which gives user feedback about how they are performing the exercises and they have done that better than their previous records and what they need to improve it on to perform better.

9 Project Application

Our project can be used in any gym or exercise based environment. A camera can be set up at a gym and the mobile application can be used to connect and get the live feed from the camera. The user can use the app to track the progress of the exercise routine and get live feedback without the need of a trainer. It can also be used for home exercise setups.