e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
Volume:02/Issue:09/September -2020 Impact Factor- 5.354 www.irjmets.com

YOGA GUIDANCE USING HUMAN POSE ESTIMATION

Prof. Sonali Muley*[1], Prateek Mahajan*[2], Pratik Medidar*[3], Harish Chavan*[4], Prashant Patil*[5]

*[1] Assistant Professor, Department of Computer Engineering, Marathwada Mitra Mandal's Institute Of Technology, Pune, India.

*[2,3,4,5] Student, Department of Computer Engineering, Marathwada Mitra Mandal's Institute Of Technology, Pune, India.

ABSTRACT

We propose a technique to almost accurately correct the human pose while performing yoga asanas. This proposed system maintains high accuracy while achieving real-time performance in almost cases. Yoga has a wide variety of asanas and here comes the role of the angle between body parts. The cosine similarity technique is used to consider the variance of the angle with original values. Critical angles come with a critical combination of angles and hence multiple dimensions need to be considered. This system finds the variance between actual and target positions and corrects the user by providing proper guidance with text to speech output in real-time to correct the selected pose. In this project we use human pose estimation for estimating the Yoga pose of the individual using computer vision technique and Open pose (open source library). All human joints will get highlighted and will be joined using greedy techniques. Euclidian angle between the body parts will be calculated and matched with the template angles, i.e. the Individual will be indicated about wrong pose using text to speech.

Keywords: Open pose, Pose net, CNN.

I. INTRODUCTION

In computer vision field localization of human body joints can be referred to as a problem of human pose estimation. For estimation of human pose there could be some of the challenges as joints could be small or barely visible and to differentiate between adjacent joints, or if there are multiple persons present in the image or video frame. The work in the field of human pose estimation is focused on detecting all possible poses, for this a variety of models have been proposed. Pose estimation could have used in animation or in fitness, it could be used in interactive installation in augmented reality. In this work, we tried to implement a pose estimation for the correction of yoga poses. Many pose detection systems are open source which requires system setup and cameras. The various model for pose estimation can detect only one person and other versions could detect multiple people. The proposed version of the model could detect a single person which is simpler.

II. METHODOLOGY

At significant level posture estimation occurs in two stages:

1. An information RGB picture is taken care of through a convolutional neural system.
2. Either a solitary posture or multi-present translating calculation is utilized to interpret presents, present certainty scores , key-point positions, and key-point certainty scores from the model yields.

At the most significant level, Pose Net will restore a posture object that contains a rundown of key-points and an example level certainty score for each identified individual. Posture certainty score decides the general trust in the estimation of a posture. It goes somewhere in the range of 0.0 and 1.0. It very well may be utilized to shroud represents that are not esteemed sufficient. key-point is a piece of an individual's represent that is assessed, for example, the nose, right ear, left knee, right foot, and so on. It contains both a position and a key-point certainty score. Pose Net at present recognizes 17 key points. Key-point certainty score decides the certainty that an expected key-point position is precise. It goes somewhere in the range of 0.0 and 1.0. It tends to be utilized to stow away key-points that are not esteemed sufficient. key-point position is x and y facilitates in the first info picture where a key-point has been distinguished. To get the key-points of the posture:

1. Image is convoluted to a smaller dimension for faster processing in the 'cmu' model for giving a higher performance in low hardware devices.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
Volume:02/Issue:09/September -2020 Impact Factor- 5.354 www.irjmets.com

2. The confidence score of each joint is returned from the model. If the confidence score is greater than a threshold of > 0.5 then it is considered as a validate joint else no joint detected is marked at that part. The confidence score ensures the presence of that part at that location in the image.
3. After inference of image, the joints are stored in the 'human' object. “humans.body_parts[i]” gives the x, y co-ordinates also the confidence score of the particular joints if required.
4. Heat map is used to determine conflict in the joint. Conflict occurs due to more than one possible location of the joints in the image. The heat map area with a higher confidence score is considered as the correct joint position in the original image.
5. Finally, for plotting a joint in the original image its x, y coordinates are obtained from the body part object in the human object and also confidence score for considering joints importance for future process.

III. MODELING AND ANALYSIS

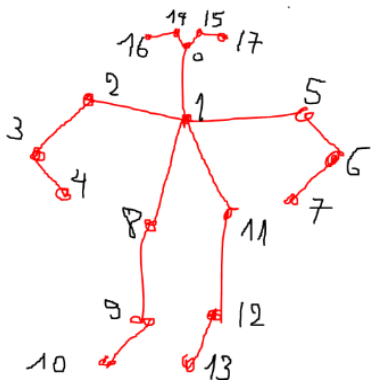

Figure-1: Keypoints mapping of humans in consideration

Almost 17 keypoint/joint of the human body which will be the reference points for this task.Let us consider the point 5,6,7 which indicate left hand of the user. If we draw a line staring from 7 till point 5 through point 6. Then point 6 acts as the midpoint forming an angle (Clockwise) which is approximately less than 90 degrees.

To Find Angle between the parts in the pose: Optimized matching strategy: Cosine distance Fig 1 shows different approaches of the same line which puts us in confusion and also the model to interpret further. To resolve this we put constraints in this as “angle < 270”. Since no joint angle between parts goes beyond 270 degrees in poses this helps us to get over angle identification.

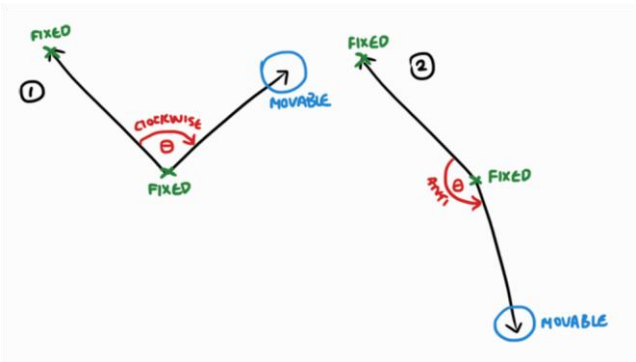

Figure-2: Clockwise and anti-clockwise angles for joint

Clockwise and anti-clockwise are the two major factors to be considered which defining angle for a joint refer Figure 2.

Therefore right and left part angles of the body are considered alternatively as required.

Confusion Matrix:

The confusion matrix will tell us the performance of the model for test data using which we can interpret certain decisions on the model.

- Positive (P): Output pose is correct.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
Volume:02/Issue:09/September -2020 Impact Factor- 5.354 www.irjmets.com

- Negative (N): Output pose is wrong. ? True Positive (TP) : Output pose is correct and input pose is also correct.
- False Negative (FN) : Output pose is wrong, but input image correct.
- True Negative (TN) : Output pose is wrong and input pose is also wrong.
- False Positive (FP) : Output pose is correct, but input pose is wrong.
- Accuracy=(TP+TN)/(TP+TN+FP+FN).

The entire system architecture can be shown in Fig.3.

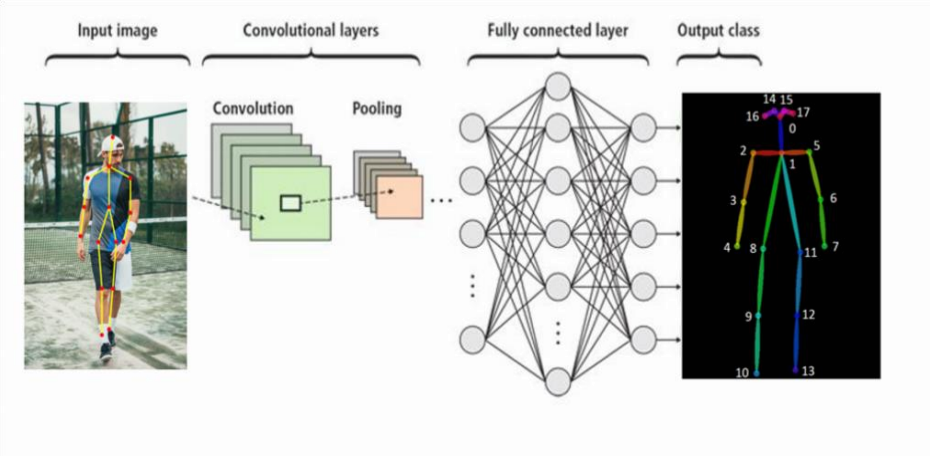

Figure-3: System Architecture

```
# USED libaries
import argparse
import logging
import time
import math
import csv

import cv2
import numpy as np

from tf_pose.estimator import TfPoseEstimator
from tf_pose.networks import get_graph_path, model_wh
```

Figure-4 : required libraries

The libraries we used for this approach are

- argparse:- Parsing arguments passed to the file.
- logging:- for logging to system, time.
- math:- Calculations for Math angle operation, csv.
- OpenCV:- display images and video in python.
- NumPy:- handling dataframes.

```
        correctAngleCount = 0
        for itr in range(8):
            angleObtained = angle(flat , angle_coordinates[itr][0],angle_coordinates[itr][1],angle_coordinates[itr][2])

            if(angleObtained<accurate_angle_list[itr]-correctionValue):
                status="more"
            elif(accurate_angle_list[itr]+correctionValue<angleObtained):
                status="less"
            else:
                status="OK"
                correctAngleCount+=1
            cv2.putText(image, angle_name_list[itr]+":- %s" % (status), (pos_onScreen[itr%2], (itr+1)*70),  cv2.FONT_HERSHEY_PLAIN, 1.5, (0, 255, 0), 2)

        posture="CORRECT" if correctAngleCount>6 else "WRONG"
        cv2.putText(image, "Posture:- %s" % (posture), (590, 80),  cv2.FONT_HERSHEY_PLAIN, 1.5, (0, 255, 0), 2)

        cv2.putText(image, "FPS: %f" % (1.0 / (time.time() - fps_time)), (590, 40),  cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
        cv2.imshow('tf-pose-estimation result', image)
        fps_time = time.time()
```

Figure-5 : computing calculations

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
Volume:02/Issue:09/September -2020 Impact Factor- 5.354 www.irjmets.com

This part tells the main Logic of the idea. Display the calculate values to the screen for the user to correct them to obtain the goal.

```
33  def angle(flat,pt1,pt2,pt3):
34      angle_calc=0
35      a = (flat[pt1*2],flat[(pt1*2)+1])
36      b = (flat[pt2*2],flat[(pt2*2)+1]) # b is midpoint
37      c = (flat[pt3*2],flat[(pt3*2)+1])
38      ang = math.degrees(math.atan2(c[1]-b[1], c[0]-b[0]) - math.atan2(a[1]-b[1], a[0]-b[0]))
39      angle_calc = ang + 360 if ang < 0 else ang
40      angle_calc = 360-angle_calc if angle_calc > 215 else angle_calc
41      return angle calc
```

Figure-6: Angle algorithm

Angle calculation Algorithm. It calculates the angle in degree unit using math function. This algorithm takes these 4 parameters input: -

- flat – It is the object of the pose which has the x , y coordinates of the joints
- pt1,pt2,pt3 – are the 3 points of the angle, of which pt2 is the midpoint.

IV. RESULTS AND DISCUSSION

Figures shows the results of detection on points and the guidance to the user towards correctness of the Pose.In fig 4 and fig 5 the points are detected and connected.

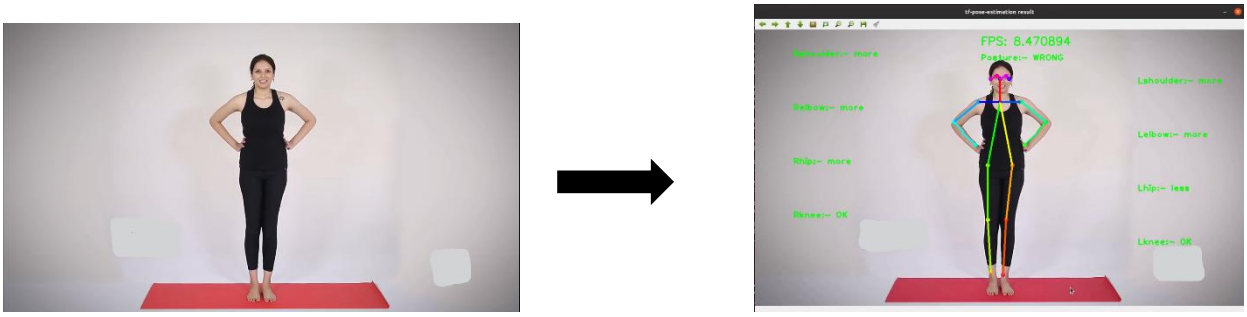

Figure-7: Initial Input and output screenshots

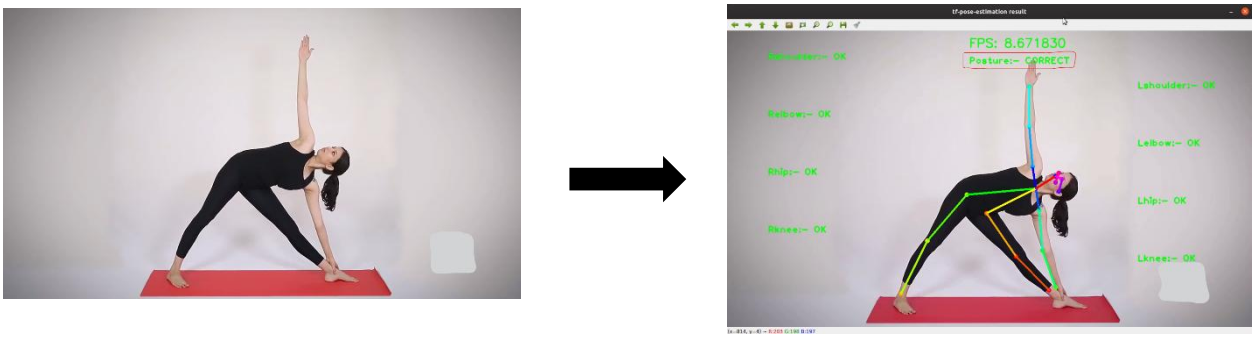

Figure-8: Final Input and output screenshots

Input video is passed to the system frame by frame . System processes each frame. For each frame 8 values are calculated Right-shoulder, Left-shoulder, Right-elbow, Left-elbow, Right-hip, Left-hip, Right-knee, Left-knee. Then after analysing the difference of these 8 obtained values the results are mapped to the screen.

V. CONCLUSION

Using Human pose identification and guidance to correction methods we implemented a real time pose corrector for yoga. System Classification accuracy varies between 75 percent to 96 percent. Depends on the Hardware specification of the system.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
Volume:02/Issue:09/September -2020 Impact Factor- 5.354 www.irjmets.com

VI. REFERENCES

[1] Zhe Cao, Shih-En Wei and Yaser Sheikh. Real-time Multi-Person 2D Pose Estimation using Part Affinity Fields. In Proc. CVPR 2017.
[2] Shih-En Wei and Varun Ramakrishna. Convolutional Pose Machines. In CVPR, Apr 2016.
[3] Alexander Toshev And Christian Szegedy. Depose: Human Pose Estimation via Deep Neural Networks. IEEE Int. Automatic Face & Gesture Recognition, Aug 2014
[4] Yu Chen, Chunhua Shen, Xiu-Shen Wei. Adversarial PoseNet: A Structure-aware Convolutional Network for Human Pose Estimation∗. 2D human pose estimation: new benchmark and state of the art analysis. In CVPR, 2015.
[5] Alejandro Newell, Kaiyu Yang and Jia Deng. Stacked Hourglass Networks for Human Pose Estimation. In CVPR Jul,2016.