IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

Smart Fitness Training System Using Computer Vision

[1] Shaik Rasool, [2] Zohaib Akhtar, [3] Sameera Begum, [4] Murtuza
[1,2,3,4] Department of Computer Science and Engineering,
[1,2,3,4] Methodist College of Engineering and Technology, Hyderabad, India

Abstract: Regular exercise is crucial for staying healthy but doing it wrong can lead to pain and injury, especially for older people. Personal trainers can be pricey, and not everyone can afford one. That is where our smart fitness training system comes in. It is like having a virtual trainer that helps you with your exercise postures, using just a webcam. We tested different computer models to see how well they recognize different exercises, and they did well. This means our system could be a reliable way to make sure you are doing your exercises right and staying safe. In addition, utilizing technology you probably already have makes it way cheaper and easier than hiring a personal trainer. Our research shows that this system could be a game-changer for keeping people healthy and happy during their workouts**.**

Index Terms - Pose Estimation, MediaPipe, Machine Learning, Fitness Training, Correct Exercise Posture.

I. Introduction

Human pose estimation is a computer vision-based technology that detects and analyses human posture. The main component of human pose estimation is the modelling of the human body. There are three of the most used types of human body models: skeleton-based model, contour-based, and volume-based. Skeleton-based model consists of a set of joints (key points) like ankles, knees, shoulders, elbows, wrists, and limb orientations comprising the skeletal structure of a human body. This model is used in both 2D and 3D human pose estimation techniques because of its flexibility [1]. Contour-based models consist of the contour and rough width of the body torso and limbs, where body parts are presented with boundaries and rectangles of a person's silhouette. Volume-based model consists of 3D human body shapes and poses represented by volume-based models with geometric meshes and shapes, normally captured with 3D scans.

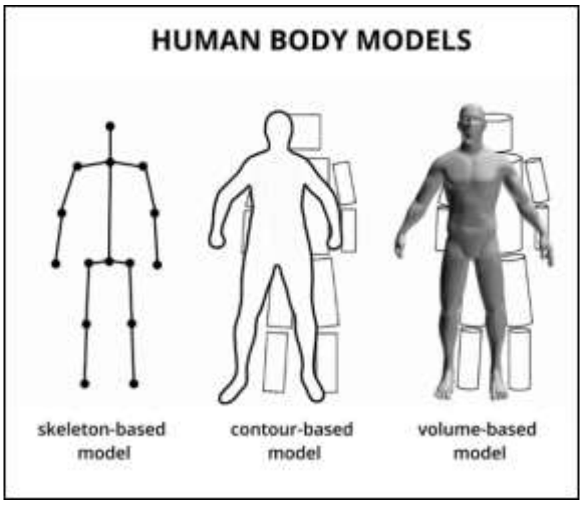


Figure 1: Human Body Model

Here, we talking about skeleton-based models, which may be detected from a 2D or 3D perspective. 2D pose estimation is based on the detection and analysis of X, Y coordinates of human body joints from an RGB image. 3D pose estimation is based on the detection and analysis of X, Y, Z coordinates of human body joints from an RGB image. When speaking about fitness applications involving human pose estimation, it is better to use 3D estimation, since it analyses human poses during physical activities more accurately. Talking about AI fitness coach apps [2].

II. RELATED WORK

2.1 Pose Detection and Tracking

Pose Detection (also known as Pose Estimation) is a widely used computer vision task that enables you to predict humans poses in images or videos by localizing the key body joints, these are elbows, shoulders, and knees, etc. MediaPipe provides a robust solution capable of predicting thirty-three 3D landmarks on a human body in real-time with high accuracy even on CPU. It utilizes a two-step machine-learning pipeline [3]. By using a detector, it first localizes the person within the frame and then uses the pose landmarks detector to predict the landmarks within the region of interest. For the videos, the detector is used only for the very first frame and then the ROI is derived from the previous frame's pose landmarks using a tracking method. In addition, when the tracker loses track of the identify body pose presence in a frame; the detector is invoked again for the next frame which reduces the computation and latency. The image below shows the thirty-three pose landmarks along with their indexes. We have learned to perform pose detection, now we will level up our game by also classifying different yoga poses using the calculated angles of various joints. We will first detect the pose landmarks and then use them to compute angles between joints and depending upon those angles we will recognize the yoga pose of the prominent person in an image. However, this approach does have a drawback that limits its use to a controlled environment; the calculated angles vary with the angle between the person and the camera. So the person needs to be facing the camera straight to get the best results. Now we will create a function that will be capable of classifying different yoga poses using the calculated angles of various joints [4].

2.2 Pose Estimation

Predicting human poses in images or videos by localising key body joints (also referred to as landmarks), such as elbows, shoulders, and knees, is a crucial aspect of pose estimation. MediaPipe provides a robust solution capable of predicting thirty-three 3D landmarks on a human body in real-time with high accuracy, even on a CPU. This is achieved through a two-step machine learning pipeline. Initially, a detector localises the person within the frame. Subsequently, the pose landmarks detector predicts the landmarks within the identified region of interest (ROI) [5[. For videos, the detector is employed only for the very first frame. Thereafter, the ROI is derived from the previous frame's pose landmarks using a tracking method. If the tracker loses track of the body pose presence in a frame, the detector is invoked again for the next frame. This approach significantly reduces computation and latency. The image below demonstrates the thirty-three pose landmarks along with their indexes.

Having learned to perform pose detection, we can now level up our application by classifying different yoga poses using the calculated angles of various joints. The process begins with detecting the pose landmarks, followed by computing the angles between joints. Based on these angles, we can recognize the yoga pose of the prominent person in an image. This approach, however, has a drawback that limits its use to a controlled environment: the calculated angles vary with the angle between the person and the camera. Therefore, for optimal results, the person needs to be facing the camera straight on [6].

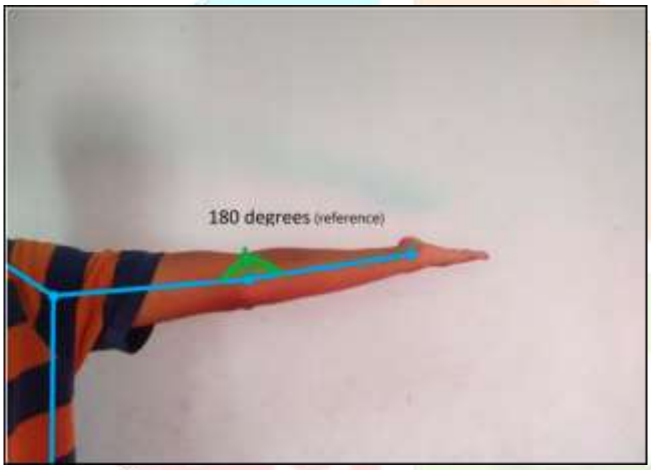
2.3 Pose Comparison

The user first selects which yoga pose he/she is going to do. When the user makes a pose, the angles made by the pose is determined and compared with the angles data of that yoga pose already available for reference in the database. When there is deviation of pose angle made by the user from the reference data, the user is notified about it in real-time via the display or audio response [7]. The rationale behind choosing angles instead of length for pose detection is because during a yoga routine the user may not always be in the center of the frame. Therefore, the measurements will keep on changing. However, if the user is assumed as the center then the angles can be measured by keeping the user as the center. Therefore, even if the user is not at the center of the frame we can calculate the angles because the angle calculation is done by keeping the user as the center point.

2.4 Feedback to the user

Giving feedback to the user is of utmost importance so that the user knows what he/she is doing wrong. This helps in guiding the user to correct posture and thus learning to practice the yoga pose correctly. The feedback regarding the performance of the user is provided in real-time via the display or audio messages. When the user deviates beyond the threshold value the user is notified. Users can observe the correction and make necessary adjustments to his/her pose to accurately practice the yoga routine. The feedback can be in the form of a visual alert on the screen or as an audio message through a headphone. So that the user need not turn his/her head to read the message on screen. Also based on the yoga pose the user may be away from the screen and may not be able correctly read the text shown on the screen. Thus, a Bluetooth connected headset or speaker is a good alternative to send feedback message to the user regarding the posture [8].

Each user has varying levels of flexibility, that is, one user may not be able to bend or flex his/her body as much as the other user. So in order to tackle that issue, a user changeable threshold parameter is included. Each user can set the threshold as per his/her requirements. A new user can set the threshold, to say 20 degrees so that he/she can have a deviation of about 20 degrees in either direction. An experienced user can set it to less than 10degrees so that he/she can practice the pose accurately [9]. This feature allows even beginner users to slowly and steadily improve his/her body flexibility to do yoga.



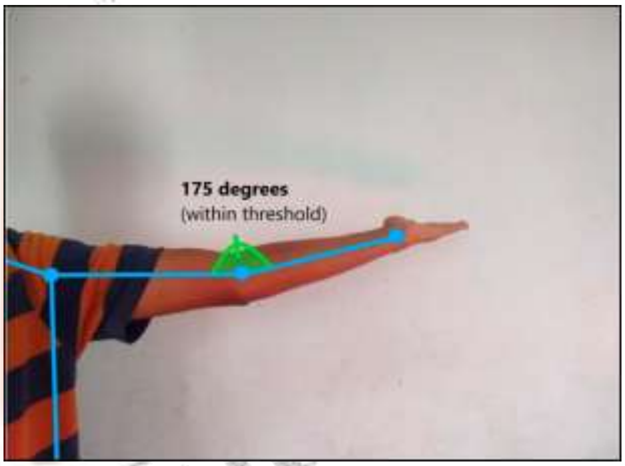


Figure 2.1 (a): Reference Angle Data Stored in the Database for Comparison

Figure 2 (b): User's Hand Pose of 175 Degrees is Within the Threshold of 5 Degrees

2.5 Human Pose Detection and Estimation

Human Pose Estimation has myriad applications that can be used in a wide variety of fields, i.e., in healthcare, sports, fitness, criminal investigation. Currently, this technology is being used for Activity recognition, Motion Capture and Augmented Reality, Motion tracking, etc. Artificial intelligence is transforming the fitness industry with new innovative technologies. In today's world, Fitness and Technology are two major industries that are proliferating [10]. The fitness industry witnessed an estimated $94 billion in 2018, with an annual growth rate of 6.1%. Whereas the AI industry of Technology dominated the market as one of the fastest growing with 38.8% of the Compound Annual Growth Rate (CAGR).

Human pose estimation is an essential technology under the computer vision community, which has been in use for the past two decades. It is a remarkable feat to understand and analyse people's gestures/posture in videos and images. With access to 1.8 billion digital photos every day, a lot of data is available to open ourselves to opportunities. Pose estimation predicts the body part or joint positions of a person from an image or a video [11].

It performs a complete biomechanics analysis of the human body in real-time; hence, it can benefit a vast number of applications. Sports analysis, Video surveillance and Assisted Living, Advanced Driver Assistance Systems (ADAS) are examples of implementations where Pose Estimation can make a difference. Human

Pose Estimation has been implemented using several approaches over the years. Initially, only the pose of a single person from an image was estimated, and this was usually done by identifying individual parts then linking these together to form a connection between the elements to create a pose [12].

III. METHODOLOGY

3.1 Initializing the Pose Detection Model

The first thing that we need to do is initialise the pose class using the ‘ mp.solutions.pose’ syntax and then we will call the setup function `mp.solutions.pose.Pose()` with the arguments:

1. static_image_mode - It is a boolean value that is if set to `False`, the detector is only invoked as needed, that is in the very first frame or when the tracker loses track. If set to `True`, the person detector is invoked on every input image. So you should probably set this value to True when working with a bunch of unrelated images not videos. Its default value is `False`.
2. min_detection_confidence - It is the minimum detection confidence with range `(0.0 , 1.0)` required to consider the person-detection model's prediction correct. Its default value is `0.5`. This means if the detector has a prediction confidence of greater or equal to 50% then it will be considered as a positive detection.
3. min_tracking_confidence - It is the minimum tracking confidence `([0.0, 1.0])` required to consider the landmark-tracking model's tracked pose landmarks valid. If the confidence is less than the set value then the detector is invoked again in the next frame/image, so increasing its value increases the robustness, but also increases the latency. Its default value is `0.5`.
4. model_complexity - It is the complexity of the pose landmark model. As there are three different models to choose from the possible values are `0`, `1`, or `2`. The higher the value, the more accurate the results are, but at the expense of higher latency. Its default value is `1`.
5. smooth_landmarks - It is a boolean value that is if set to `True`, pose landmarks across different frames are filtered to reduce noise. But only works when `static_image_mode` is also set to `False`. Its default value is `True`. Then we will also initialize `mp.solutions.drawing_utils` class that will allow us to visualize the landmarks after detection, instead of using this, you can also use OpenCV to visualize the landmarks [13].

3.2 Perform Pose Detection

Now pass the image to the pose detection machine learning pipeline by using the function `mp.solutions.pose.Pose().process()`. But the pipeline expects the input images in `RGB` color format so first we will have to convert the sample image from `BGR` to `RGB` format using the function [`cv2.cvtColor()`] as OpenCV reads images in `BGR` format (instead of `RGB`) [14].

After performing the pose detection, we will get a list of thirty-three landmarks representing the body joint locations of the prominent person in the image. Each landmark has:

`x` - It is the landmark x-coordinate normalized to [0.0, 1.0] by the image width.

`y`: It is the landmark y-coordinate normalized to [0.0, 1.0] by the image height.

`z`: It is the landmark z-coordinate normalized to roughly the same scale as `x`. It represents the landmark depth with midpoint of hips being the origin, so the smaller the value of z, the closer the landmark is to the camera.

visibility: It is a value with range [0.0, 1.0] representing the possibility of the landmark being visible (not occluded) in the image. This is a useful variable when deciding if you want to show a particular joint because it might be occluded or partially visible in the image.

3.3 Creating a Pose Detection Function

This function performs pose detection on an image.

1. Args:

Image: The input image with a prominent person whose pose landmarks needs to be detected.

Pose: The pose setup function required to perform pose detection.

Display: A Boolean value that is if set to true the function displays the original input image, the resultant image, and the pose landmarks in 3D plot and returns nothing.

2. Returns:
   output_image: The input image with the detected pose landmarks drawn.
   landmarks: A list of detected landmarks converted into their original scale.

After performing the pose detection, we will get a list of thirty-three landmarks representing the body joint locations of the prominent person in the image.

Each landmark has:
   \`x\` - It is the landmark x-coordinate normalized to [0.0, 1.0] by the image width.
   \`y\`: It is the landmark y-coordinate normalized to [0.0, 1.0] by the image height.
   \`z\`: It is the landmark z-coordinate normalized to roughly the same scale as \`x\`. It represents the landmark depth with midpoint of hips being the origin, so the smaller the value of z, the closer the landmark is to the camera.
visibility: It is a value with range [0.0, 1.0] representing the possibility of the landmark being visible (not occluded) in the image. This is a useful variable when deciding if you want to show a particular joint because it might be occluded or partially visible in the image.

After performing the pose detection on the sample image above, we will display the first two landmarks from the list, so that you get a better idea of the output of the model [15].

3.4 Pose Classification with Angle Heuristics

We have learned to perform pose detection, now we will level up our game by also classifying different yoga poses using the calculated angles of various joints. We will first detect the pose landmarks and then use them to compute angles between joints and depending upon those angles we will recognise the yoga pose of the prominent person in an image [16].

This function classifies yoga poses depending upon the angles of various body joints.

1. Args:
   landmarks: A list of detected landmarks of the person whose pose needs to be classified.
   output_image: A image of the person with the detected pose landmarks drawn.
   display: A boolean value that is if set to true the function displays the resultant image with the pose label written on it and returns nothing.

2. Returns:
   output_image: The image with the detected pose landmarks drawn and pose label.
   written.label: The classified pose label of the person in the output_image.

3.5 Pose Detection on Real-Time Webcam Feed/Video

The results on the images were pretty good, now we will try the function on a real-time webcam feed and a video. Depending upon whether you want to run pose detection on a video stored in the disk or on the webcam feed, you can comment and uncomment the initialization code of the Video Capture object accordingly [17].

IV. CONCLUSION

Smart fitness training system holds promise as a game-changer in personal fitness, promoting safer, healthier, and more efficient workouts. Future work will focus on further refining the system, expanding its exercise repertoire, and enhancing its user interface to ensure it meets the diverse needs of users. Our research indicates that the system's high accuracy in recognizing various exercises can reliably assist users in their workouts, making it a valuable tool for individuals, especially older adults, who may be more prone to exercise-related injuries. The development and testing of our smart fitness training system demonstrates its potential as an effective and affordable solution for promoting proper exercise techniques and preventing injuries. By leveraging human pose estimation technology with commonly available webcams, the system offers real-time feedback on exercise postures, ensuring users maintain correct form. This approach not only makes personal fitness training more accessible but also significantly reduces the need for expensive personal trainers.

REFERENCES

[1] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black. Keep it on the SMPL:Automatic estimation of 3d human pose and shape from a single image. Available: European Conference on Computer Vision (ECCV), pages 561–578.Springer https://www.paperdigest.org[Accessed : March 2016].

[2] Rasool, Shaik, and U. N. Dulhare. "Eco-friendly auto lacing smart shoes for fitness and facile navigation using IoT." (2021): 261-266.

[3] Dulhare U.N., Rasool S. (2022) Smart Airport System to Counter COVID-19 and Future Sustainability. In: Satyanarayana C., Gao XZ., Ting CY., Muppalaneni N.B. (eds) Machine Learning and Internet of Things for Societal Issues. Advanced Technologies and Societal Change. Springer, Singapore. https://doi.org/10.1007/978-981-16-5090-1_5

[4] Pauwels, Karl, Leonardo Rubio, and Eduardo Ros. "Real-time pose detection and tracking of hundreds of objects." IEEE Transactions on Circuits and Systems for Video Technology 26, no. 12 (2015): 2200-2214.

[5] C.-H. Chen and D. Ramanan. 3D human pose estimation = 2D pose estimation +matching.Available : Conference on Computer Vision and Pattern Recognition(CVPR), pages 5759 5767. https://arxiv.org/abs/

[6] E. Brau and H. Jiang. 3d human pose estimation via deep learning from 2d. annotations. Available : In International Conference on 3D Vision (3DV), pages 582–591.IEEE

[7] Dulhare, U.N., Rasool, S., Kumar, G., Khan, G., Gunjan, V.K. (2023). A Novel Approach for Speech Emotion Recognition with Facial Expression Analysis. In: Kumar, A., Gunjan, V.K., Hu, YC., Senatore, S. (eds) Proceedings of the 4th International Conference on Data Science, Machine Learning and Applications. ICDSMLA 2022. Lecture Notes in Electrical Engineering, vol 1038. Springer, Singapore. https://doi.org/10.1007/978-981-99-2058-7_25

[8] Gepperth, Alexander, Michael Garcia Ortiz, and Bernd Heisele. "Real-time pedestrian detection and pose classification on a GPU." In 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), pp. 348-353. IEEE, 2013.

[9] Shaik Rasool, Uma N Dulhare, “Data Center Security”, “Green Computing in Network Security: Energy Efficient Solutions for Business and Home “, (1st ed.). CRC Press, New York, 11 January 2022, ISBN: 9781003097198, DOI: https://doi.org/10.1201/9781003097198

[10] Wang, Jiang, Zicheng Liu, and Ying Wu. Human action recognition with depth cameras. Berlin, Germany:: Springer, 2014.

[11] Yadav, Santosh Kumar, Amitojdeep Singh, Abhishek Gupta, and Jagdish Lal Raheja. "Real-time Yoga recognition using deep learning." Neural computing and applications 31 (2019): 9349-9361.

[12] Prince, Simon JD. Computer vision: models, learning, and inference. Cambridge University Press, 2012.

[13] Poirson, Patrick, Phil Ammirato, Cheng-Yang Fu, Wei Liu, Jana Kosecka, and Alexander C. Berg. "Fast single shot detection and pose estimation." In 2016 Fourth International Conference on 3D Vision (3DV), pp. 676-684. IEEE, 2016.

[14] Kumar, M. Kiran, Shaik Rasool, and S. Jakir Ajam. "Web data mining using xML and agent framework." IJCSNS 10, no. (2010): 175.

[15] Dulhare, Uma N., and Shaik Rasool. "Digital Evidence in Practice." (2016).

[16] Bazarevsky, Valentin, and Ivan Grishchenko. "On-device, real-time body pose tracking with mediapipe blazepose." Google AI Blog (2020).

[17] Wei, Shih-En, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. "Convolutional pose machines." In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 4724-4732. 2016.