Yoguru – Computer-based Yoga Assistant

Hai Nguyen
Bachelor of Data Science
SP Jain School of Global Management
Sydney, Australia
hai.bj19bds003@spjain.org

Rithwik Chhugani
Bachelor of Data Science
SP Jain School of Global Management
Sydney, Australia
rithwik.bs18bds008@spjain.org

***Abstract*— There has been an increasing trend of self-training for health benefits and Yoga is by far considered as one of the most effective form of exercises to enhance both mental and physical health. Researchers have proven that Yoga could be a great cure for many diseases, mental illness. Yoga exercises help boost physical health as well as cleanse human body, mind and soul. YoGuru is a project that helps recognize yoga exercise activities and gives suggestions for your postures to maximize health benefits and minimize potential injuries with the use of deep learning-based image classification and postures recognition model.**

Keywords—yoga, OpenPose, Pose estimation, Deep learning, Machine learning, AI

I. INTRODUCTION

In the modern life, people are becoming more and more concerned about their health and fitness. Training is proved to help people in both physical and mental health [1], which is a great way to remain a balanced lifestyle. It is a common practice to have a personal trainer who could monitor the performance and give guidance towards the maximum gains while practicing. This is also the case for Yoga, where postures are considered everything that matters. To avoid any misinterpretation and potential risks during trainings, participants usually look for Yoga teachers who are highly qualified in the field as their personal trainers. Finding a qualified and available trainer, however, is not always an easy task considering the proximity of accessibility, budget and quality.

The mentioned difficulties can be eliminated with the help of AI and modern technology approaches. Yoguru is an application model functioning as a personal trainer for those who want to practice yoga at their own conditions. The deep learning-based model will detect participants postures and give comments in real time. This virtual trainer will make sure that all the body parts are contributing to the final best performance that optimize the gains. The use of computer and technology power will ensure the precision and unbiases when it comes to redundant tasks and especially various factors that may affect the training procedures. With mobile devices connected to the internet, participants can start working on their own fitness goals at their own pace and state, which helps make life become easier and more convenient..

II. LITERATURE REVIEW

A. Yoga History

According to Dr. Ishwar V. Basavaraddi, yoga is believed to be a practice that has started at the beginning of the civilization and rooted in India [2]. Yoga is a posture-based practice and there are several variations of yoga-like activities available. The most common yoga practice has 84 unique postures, which are also called asanas. Yoga combines the focus on self-awareness, the breath, energy and muscular activities, which makes it a form of mind-body fitness practice [3]. Yoga has been gaining more popularity all over the world and is recognized by the National Institutes of Health as a form of Complementary and Alternative Medicine (CAM) [4]. Researchers have pointed out the benefits of yoga practice in enhancing both people mental and physical health. Yoga exercises help boost physical health as well as cleanse human body, mind and soul. Talking about peaceful state of mind, it is considered as a secret weapon to battle against the stress and difficulties during the current COVID-19 pandemic. Everyone is trying to do something or the other to keep calm and make the most out of this current situation. In the US, 21st June has been declared as the "International Day of Yoga" since 2014 [5] Estimating the pose of different asanas is a challenging task as it involves complicated configurations of postures. In this project, we will work on two asanas namely Warrior II and Extended Hand-To-Big-Toe

1. *Warrior II Pose:*

Warrior II, also known as Virabhadrasana II, is one of the most common poses in yoga practice. This exercise requires practitioners to make use of all the body parts in a combination of actions (as shown in figure 1). Trainers need to open their hips and shoulders while stretching the inner thighs, groin and chest at the same time. This exercise helps strengthen the legs, abs and arms.

2. *Extended Hand-To-Big-Toe Pose:*

This is an exercise focusing on maintaining the solid grounding through the standing foot that helps keep the individual steady. In this pose, trainers are required to stand on one leg while stretching the other one to the side and touching its big toe with the right arm. (as shown in figure 2). This extended hand-to-big-toe exercise helps strengthen the legs, ankles and improve the sense of balance.

Figure 1: Warrior II pose

Figure 2: Extended hand-to-big-toe pose

B. Self-Training With Technology Assistance Scenario:

Health-related activities, especially sports and exercises have long been of interest to the general public. With the advancement in technology in general and the outbreak of artificial intelligent in specific, scientists and researchers have moved forward in developing tools and technologies that help athletes and trainers do their best in practice. For example, in soccer, Zhu et al. (2009) proposed a system that analyses the interaction between the ball and the players using spatial-temporal data from streaming video to identify the tactical patterns [6]. Or like Chen et al. (2012) work on evaluating pitching in basketball, estimating shooting location in volleyball by tracking the ball motions in videos and providing its trajectory [7].

A great number of these works focus on proposing automated or semi-automated systems specialized in self-training that analyze and assist participants to improve their performance and avoid injuries in sports. With posture-based exercises like Yoga, it is very important to keep it right to avoid any potential injuries during practice. To address this problem, Patil et al. have proposed his "Yoga Tutor" project to detect the difference between the postures of trainers and professionals using the speeded up robust features (SURF) [8]. However, this model was considered unable to make appropriate comparison since it only uses the contour information. Luo et al. introduced Yoga training system which utilized the Motion Replication Technique (MoRep) and an interface consisting of 16 Inertial Measurement Units (IMU) and 6 tactors, which directly affect how participants exercise [9]. An image and text-based model is also proposed by Wu et al. to instruct trainers but it fails to analyze their postures [10].

Chen et al. introduced a computer assisted self-training system [11] that uses star skeleton approach to acquire the descriptor for trainers' poses. The project can classify 3 major postures: 'tree, warrior III and downward facing dog' at 82.84% of accuracy. Similar work with Deep learning approach was introduced by Shruti which can classify up to 6 yoga asanas at nearly perfect accuracy. Trejo et al. also proposed a model that classify 6 postures using Kinect depth camera, which is not generally available [12].

Using image recognition techniques, Mohanty et al. proposed a Yoga postures identifications system that make use of convolutional neural network (CNN) and stacked autoencoder (SEA) algorithms [13]. The shortcoming of this system lies in the model that just process image input, not streaming video.

Since conventional skeleton extraction approaches involve high computational cost and are sensitive to noise, deep learning approaches have been introduced to replace them. DeepPose was one of the pioneers since it utilized the neural network regressors to do regression on coordinates of joints. It also gives prediction for the hidden parts of body to increase the model accuracy. However, this approach was limited by the localize problem.

C. Pose Estimation Approaches

1. MaskRCNN:

MaskRCNN is an instance-based segmentation model. Suppose we have a satellite image which has buildings in it, a semantic segmentation model will be able to identify the difference between the buildings and the ground, but the instance-based segmentation model will also mark boundaries for different buildings shown in the image. The MaskRCNN model is developed on top of the Faster R-CNN model [14]. This model is quite powerful but doesn't fit well with needs of the problem statement. MaskRCNN is more applicable in regions where the problem statement revolves around identifying different objects in the image, classifying them and imposing a mask on the identified object.

2. AlphaPose:

The AlphaPose algorithm follows the RMPE framework. It is SPPE-based and follows the two-step framework which contains human detection and pose estimation. The whole framework mainly consists of three blocks: Symmetric Spatial Transformer Network (SSTN), Parametric Pose Non-Maximum Suppression (NMS) and Pose-Guided Proposals Generator (PGPG). The NMS used here is like the one used in OpenPose. The SSTN attached to the SPPE is used for obtaining an accurate single person region from a rough bounding box. After detecting human proposals, SSTN transforms the proposals to make them more suitable for SPPE, and de-transforms them after SPPE [15]. To improve this step, in the training process, there is an added parallel SPPE branch which backpropagates the center-located pose errors. A parametric pose NMS is then introduced to eliminate redundant pose estimations (calculating peaks), which defines a novel pose distance metric with a data-driven optimizer. Finally, the PGPG is employed for data augmentation by learning the output distribution of human detection results.

3. DeepCut:

Experimental studies have highlighted a unique comparison metric to differentiate between use of OpenPose and AlphaPose. It's observed that AlphaPose shows varying accuracy which is dependent on the number of people present in the frame whereas the OpenPose is found to be independent of such issues [16].

D. OpenPose Library:

OpenPose was introduced by Perceptual Computing Lab of Carnegie Mellon University as the first real-time multi-person system that jointly detect and describe the keypoints of human body, face, hand and foot on a single image. This library used the CMU Panoptic Studio dataset as training data and multi-stage convolutional neural networks (CNNs) architecture to identify a total of 135 joint keypoints on human body [17]. Convolutional neural networks (CNNs) architecture is a state-of-the-art deep learning approach that

is widely used in computer vision since it outperformed many other traditional machine learning approaches when it comes to visual recognition tasks [18].
OpenPose librry will detect human body joints using an RGB camera. In this project, the library will use the COCO output format which detects 17 keypoints including: eyes, ears, neck, nose, elbows, shoulders, knees, wrists, ankles and hips.

Figure 3: COCO Output Format

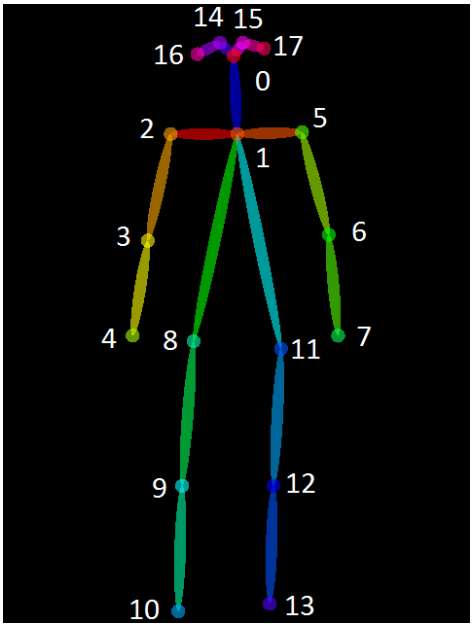


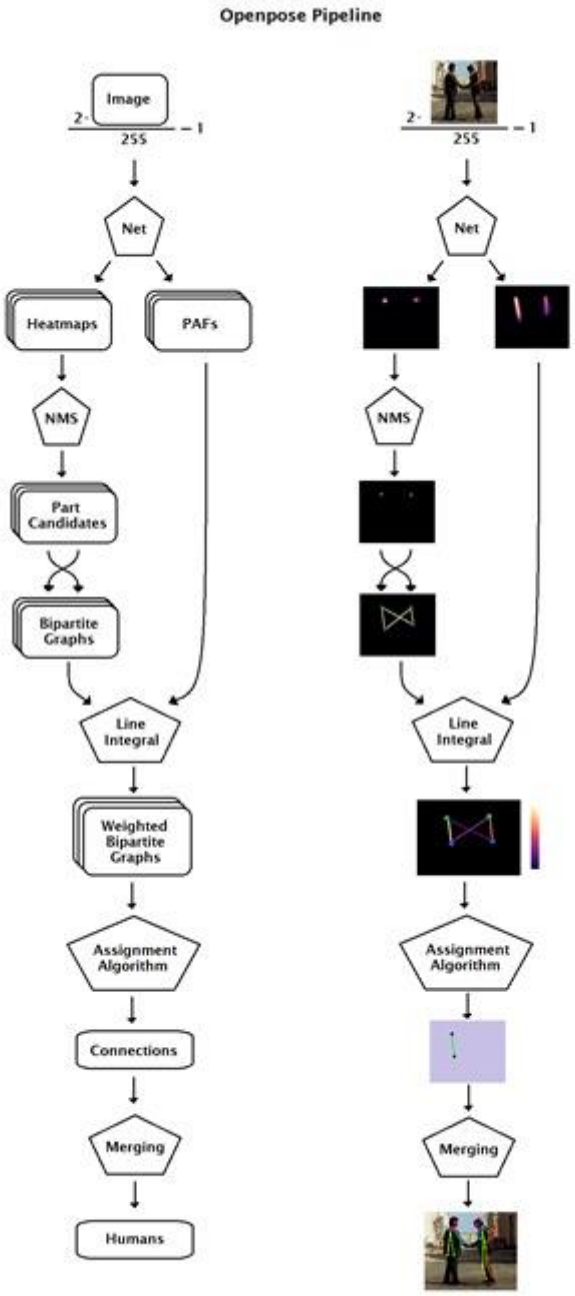
Table 1: COCO Output Keypoints

No	*Keypoint*	*No*	*Keypoint*
0	Nose	9	Right knee
1	Neck	10	Right foot
2	Right shoulder	11	Left hip
3	Right elbow	12	Left knee
4	Right wrist	13	Left foot
5	Left shoulder	14	Right eye
6	Left elbow	15	Left eye
7	Left wrist	16	Right ear
8	Right hip	17	Left ear

At the first stage, OpenPose will detect the keypoints of each individual in the image and then assign these parts to each person. The pipeline of the OpenPose is described in the figure 4 and 5 as follows

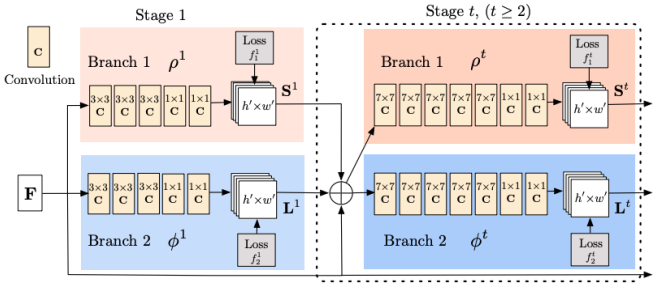
- Network: Frist 10 layers of the VGG19 network
- Heatmaps: Matrices showing probable location of body parts
- PAFs: Matrices that store locations and orientations of pairs
- NMS: Similar to maxpooling with a window size of 5 used to find peaks in the matrix
- Bipartite Graphs: Used to make logical connections of body parts and also help in differentiating between body parts of different people
- Line Integral: Combines PAFs and bipartite graphs
- Assignment Algorithm: Take the output of the line integrals, sorts the matrix and selects unique combinations with high edge values
- Merging: Merging all points and imposing the stick figure on the body

Figure 4: OpenPose Pipeline



OpenPose network starts by extracting image features using the initial VGG-19 layers. While the first branch will give prediction of 18 confidence maps representing specific parts of the body, the second branch predicts the degree between parts, which is denoted as 38 Part Affinity Fields (PAF) (as shown in figure 5).

Figure 5: Multi-stage CNNs Architecture



The predictions from previous branches are then refined by sequence of stages. In the next step, OpenPose will base on confidence maps from previous step to form the bipartite graph. PAF values will be used to eliminate the weaker links in those graphs. At the end, the body keypoints structure are estimated for each individual captured in the image.

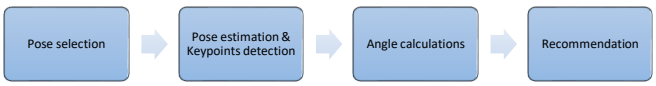
III. Methodology

In this project, the deep learning framework tensorflow, version 2.4.0 was used to execute the OpenPose library for pose estimation. The model is also compatible with computers without Graphics Processing Unit (GPU) coming with the limitation in performance including speed and model accuracy. The prototype of this model was run on computer with following configurations:

Table 2: Machine Configuration

OS	Microsoft Windows 10 Home Single Language
Processor	Intel (R) Core i5-7200U CPU @ 2.50GHz
GPU supported	No
RAM	4GB
Language used	Python
Version	3.8.2

The overall pipeline of the application works as follows:

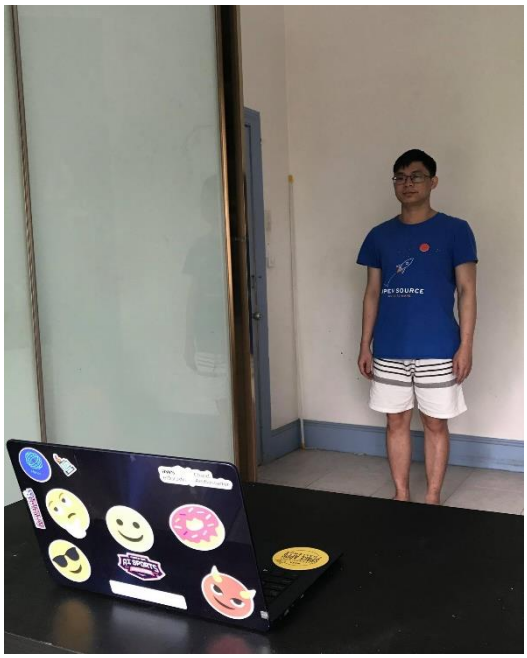


First, the trainer must choose which pose to practice, in this project, 2 poses are offered, the Warrior II and the Extended Hand-To-Big-Toe. The model will take real-time input from users' camera then feed this streaming data to the OpenPose deep learning model for pose estimation. At this stage, the model will detect and estimate different keypoints and draw a map of the body skeleton. Based on the coordinate of the detected keypoints, the model will calculate the corresponding angles of each defined pose. These angles will then be compared to the predefined figures and make recommendations for any adjustment. These comments will appear on screen to instruct the trainer to practice correctly.

A. Input Data

Trainer has to stand in front of the camera at a distance of 3-4 meters to make sure that the whole body is visible in the frame. There must be only one individual in front of the camera during the training session to ensure the accurate recommendation. The streaming video is taken at the rate of 30 FPS (frames per second).

Figure 6: Input demo



B. Angle Calculation Function

To calculate the angles between parts of the body, we will use the inverse cosine function with the python implementation shown as follows:

```
angle = math.acos( (a+b-c) / math.sqrt(4*a*b) ) * 180/math.pi
```

The angles will be calculated using the defined "angle_calc" function. This function takes 3 arguments which are the keypoints detected from the "find_point" method and takes the coordinate values for math calculations. When finishes, the function will return the corresponding angle for further use.

For the Warrior II pose, we calculate 4 angles, namely left hand, left leg, right hand, right leg. For the Extended Hand-To-Big-Toe pose, we calculate 1 extra angle between right hand and right leg due to the requirement of the pose. The keypoints used to calculate these angles are shown in table 2 as follows.

Table 3: Angles and Corresponding Keypoints

Angle	**Set of keypoints**
Right hand	2,3,4
Left hand	5,6,7
Right leg	8,9,10
Left leg	11,12,13
Right hand and right leg	1,8,9

C. *Recommendation configurations*

The angle configurations for each pose are different due to the different representations. Since the angles are estimated, the measure of each angle will be considered in a flexible manner by accepting values belong to the acceptable range.

The range for each angle used in this project is shown in table 4.

Table 4: Acceptable range for angles

	Warrior II	**Extended Hand-To-Big-Toe**
Right hand	170-180	170-180
Left hand	170-180	170-180
Right leg	80-90	80-90
Left leg	170-180	170-180
Right hand and right leg	Not applicable	60-70

If all the angles fall into the acceptable range, the model will consider the pose as perfectly correct. In case one angle cannot be calculated or does not lie in the acceptable range, the corresponding recommendation will be given and output to display on screen.

Table 5: Recommendations for Warrior II Pose

Condition	**Recommendation**
Right hand < 170	You should straighten up your right forearm
Left hand < 170	You should straighten up your left forearm
Right leg > 90	You should get lower on your right leg
Left leg < 170	You should straighten up your left leg

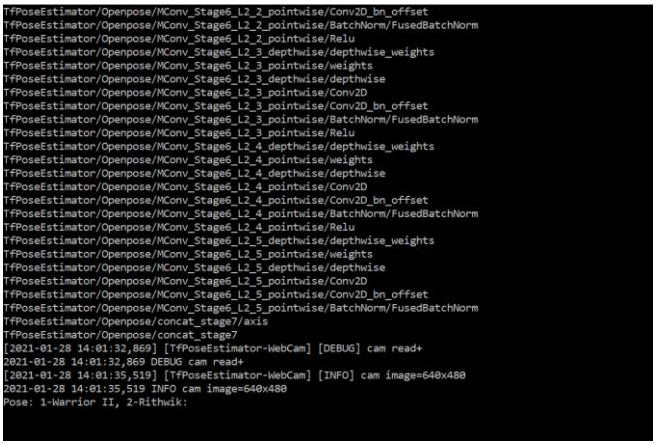
Table 6: Recommendations for Extended Hand-To-Big-Toe Pose

Condition	**Recommendation**
Right hand < 170	You should straighten up your right forearm
Left hand < 170	You should straighten up your left forearm
Right leg > 170	You should straighten up your right leg
Left leg < 170	You should straighten up your left leg
Right leg and hand > 70	You should raise your right leg more till you touch your feet

IV. IMPLEMENTATION AND RESULTS

A. *Mode selection*

Figure 7: Mode selection



Practitioners choose the pose they want to practice by inputting the corresponding pose number.

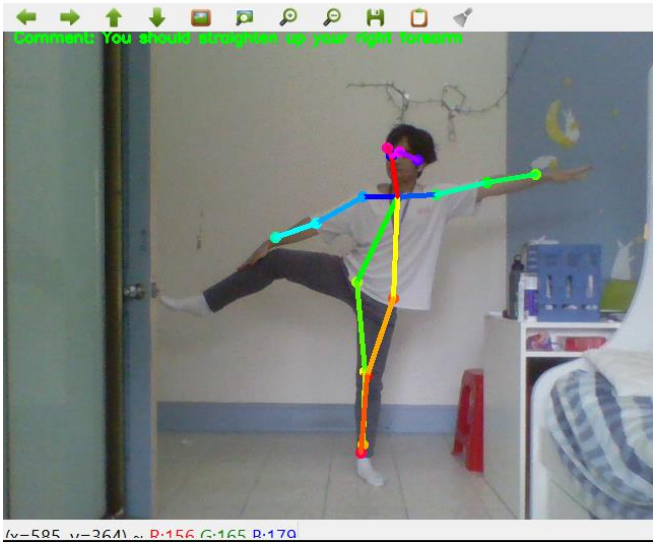
1. Warrior II
2. Extended Hand-To-Big-Toe

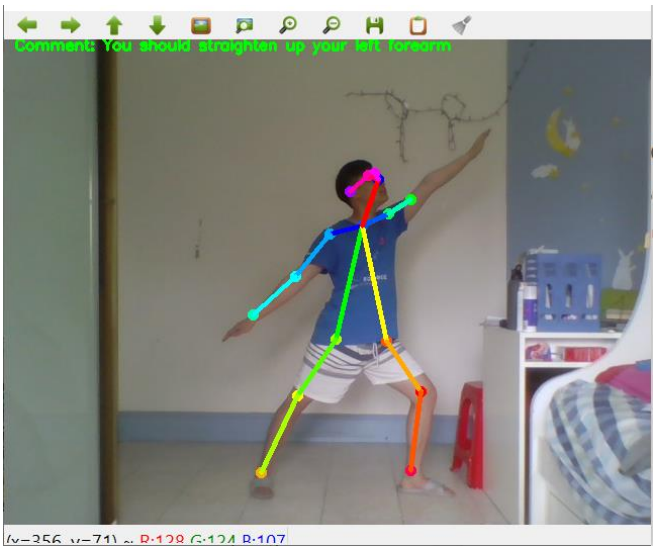
B. *Real-time pose estimation*

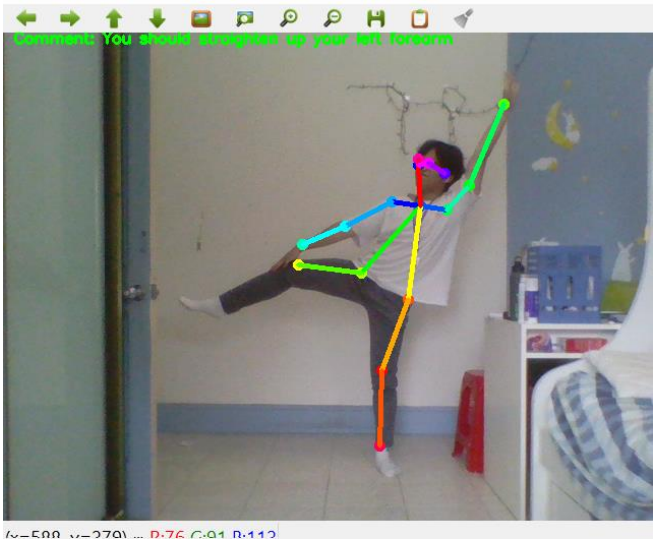
In general, the model is good at estimating the pose and detecting all the keypoints to draw the body skeleton map in most cases. The accuracy in keypoints detection results in more accurate calculations for the recommendation system, which returns the suggestions for the trainers to correct their pose during practice.

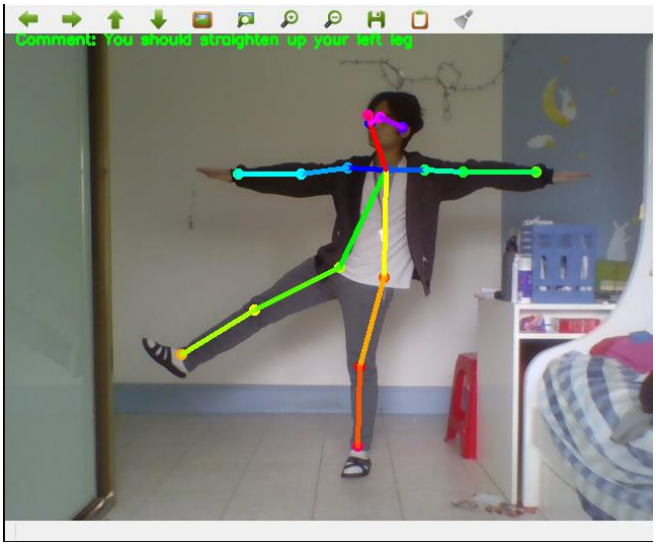
For the Warrior II pose, the model can detect all the keypoints with an accuracy of 99%. On the contrary, the model struggles with detecting the raised right leg during practice, which results in less accurate recommendations during practice as shown in Fig. 8.

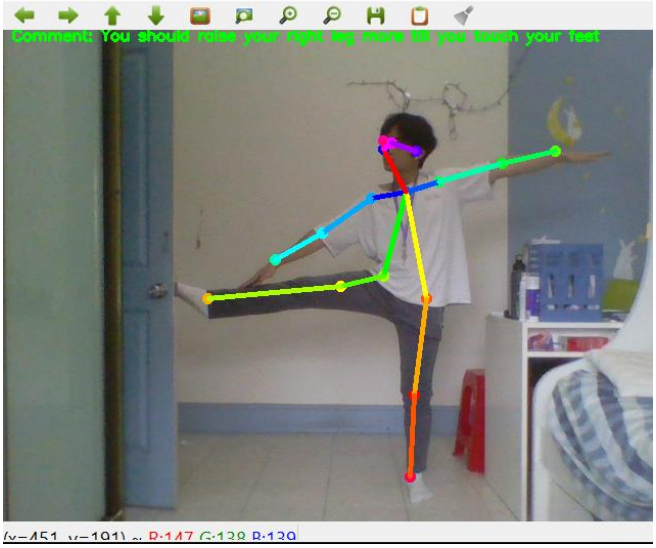
Figure 8: Pose estimation in real time.











V. LIMITATION AND FUTURE WORKS

The app is currently optimized for individuals and not groups. People may want to practice group yoga sessions to maintain the bond and joy of being together. The addition of the group yoga sessions could be integrated for the future version of the app. The current application only works on 2

predefined poses, which can be extended to multiple poses in future works.

For the prototype the app, gives suggestion in form of text on the screen which could later be converted to voice feedback to improve the overall functionality of the app.

In future work, hybrid models or different algorithm can be used to solve the problem of detecting the raised right leg appearing during this experiment.

As of now the app is fed with two basic yoga postures to demonstrate the capabilities of it. In future the app could have two separate versions: one for the user and the other one for a yoga professional. The user side of the app remains the same, but the addition of the yoga professional version of the app will make it easier for professionals to add more poses for people to practice.

YoGuru can be transformed in form an app for iOS/Android or developed as a web app.

REFERENCES

[1] Sharma, A., Madaan, V., & Petty, F. D. (2006). Exercise for mental health. *Primary care companion to the Journal of clinical psychiatry*, *8*(2), 106. https://doi.org/10.4088/pcc.v08n0208a J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] V. Basavaraddi, I., 2015. Yoga: Its Origin, History And Development.

[3] Yoga: intuition, preventive medicine, and treatment. Collins C, J Obstet Gynecol Neonatal Nurs. 1998 Sep-Oct; 27(5):563-8.

[4] Williams K, Steinberg L, Petronis J. Therapeutic application of iyengar yoga for healing chronic low back pain. Int J Yoga Ther. 2003;13:55–67

[5] Guddeti RR, Dang G, Williams MA, Alla VM (2018) Role of Yoga in cardiac disease and rehabilitation. J Cardiopulm Rehabil Prev.

[6] Zhu, Guangyu & Xu, Changsheng & Rui, Yong & Jiang, Shuqiang & Gao, Wen & Yao, Hongxun. (2009). Event Tactic Analysis Based on Broadcast Sports Video. IEEE Transactions on Multimedia. 11. 49-67. 10.1109/TMM.2008.2008918.

[7] Hua-Tsung Chen, Chien-Li Chou, Tsung-Sheng Fu, Suh-Yin Lee, Bao-Shuh P. Lin,Recognizing tactic patterns in broadcast basketball video using player trajectory, Journal of Visual Communication and Image Representation, Volume 23, Issue 6, 2012, Pages 932-947, ISSN 1047-3203, https://doi.org/10.016/j.jvcir.2012.06.003.

[8] Patil S, Pawar A, Peshave A et al (2011) Yoga tutor: visualizationand analysis using SURF algorithm. In: Proceedings of 2011IEEE control system graduate research colloquium, ICSGRC2011, pp 43–46.

[9] Z. Luo et al., "“Left Arm Up!” Interactive Yoga training in virtual environment," 2011 IEEE Virtual Reality Conference, Singapore, 2011, pp. 261-262, doi: 10.1109/VR.2011.5759498.

[10] W. Wu, W. Yin and F. Guo, "Learning and Self-Instruction Expert System for Yoga," 2010 2nd International Workshop on Intelligent Systems and Applications, Wuhan, 2010, pp. 1-4, doi: 10.1109/IWISA.2010.5473592.

[11] Chen HT, He YZ, Hsu CC et al (2014) Yoga posture recognitionfor self-training. In: Lecture notes in computer science (includingsubseries lecture notes in artificial intelligence and lecture notesin bioinformatics), pp 496–505

[12] E. W. Trejo and P. Yuan, "Recognition of Yoga Poses Through an Interactive System with Kinect Device," 2018 2nd International Conference on Robotics and Automation Sciences (ICRAS), Wuhan, 2018, pp. 1-5, doi: 10.1109/ICRAS.2018.8443267.

[13] Mohanty A., Ahmed A., Goswami T., Das A., Vaishnavi P., Sahay R.R. (2017) Robust Pose Recognition Using Deep Learning. In: Raman B., Kumar S., Roy P., Sen D. (eds) Proceedings of International Conference on Computer Vision and Image Processing. Advances in Intelligent Systems and Computing, vol 460. Springer, Singapore. https://doi.org/10.1007/978-981-10-2107-7_9

[14] Kaiming He, Georgia G., Piotr Dollár and Ross Girshick, “Mask R-CNN”, arXiv:1703.06870 [cs] Mar. 2017.

[15] H. Fang, S. Xie, Y. Tai and C. Lu, "RMPE: Regional Multi-person Pose Estimation," 2017 IEEE International Conference on Computer Vision (ICCV), Venice, 2017, pp. 2353-2362, doi: 10.1109/ICCV.2017.256.

[16] L. Pishchulin et al., "DeepCut: Joint Subset Partition and Labeling for Multi Person Pose Estimation," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 4929-4937, doi: 10.1109/CVPR.2016.533.

[17] Z. Cao, G. Hidalgo, T. Simon, S. -E. Wei and Y. Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 1, pp. 172-186, 1 Jan. 2021, doi: 10.1109/TPAMI.2019.2929257.

[18] Salman Khan; Hossein Rahmani; Syed Afaq Ali Shah; Mohammed Bennamoun; Gerard Medioni; Sven Dickinson, A Guide to Convolutional Neural Networks for Computer Vision, Morgan & Claypool, 2018, doi: 10.2200/S00822ED1V01Y201712COV015.