Acknowledgement

It gives us immense pleasure to express our deepest sense of gratitude and sincere thanks to all those whose motivation and encouragement has made the completion of this project possible. Thanks, and gratitude are due to a lot of people without whom none of this would have been possible. We must express our sincere gratitude towards the Kathmandu Engineering College and Department of Computer Engineering of Kathmandu Engineering College. We would like to acknowledge our deep sense of gratitude to our Head of Department **Er. Sudeep Shakya**, Deputy Head of Department of Computer Engineering **Er. Kunjan Amatya** and our project coordinator **Er. Sharad Chandra Joshi** for giving us this opportunity to undertake this project.

We would especially like to thank our supervisor **Er. Sharad Neupane** for his wholehearted support. We are also grateful to our yoga teacher **Mr. Shanta Kharel** for his continuous support.Also we would like to thank our friends **Anurag Shukla , Diwash Adhikari** and **Nikita Masal** for helping us in data collection. Last but not the least we would like to thank all our friends and other staffs who helped to complete this project and give our sincere gratitude towards anyone who helped us directly or indirectly to finalize this report.

Abstract

Incorrect posture while performing yoga can lead to serious injury to the body. Thus to prevent this we present an intuitive approach based on keypoints detection and Long Short Term Memory to correct the practitioner's pose while performing various yoga asanas. The system can identify the asana performed by the practitioner in real time and also provides a feedback to decrease the chances of any injuries and increases the knowledge of yoga. A data-set has been created of six yoga asana (i.e., Bhujangasana, Vrksasana, Virabhadrasana, Tadasana, Shavasana, Padmasan) by capturing video from different individuals who have been involved in the project under the supervision of a Yoga teacher. We have used a deep learning model which uses mediapipe pose model to extract keypoints from different frames of the video followed by Recurrent Neural Network, Long short-term memory to give a temporal prediction on which asana is being performed. The system also provides feedback on yoga pose by extracting keypoints and calculating the angle of the joints.

Keyword: *Yoga, Deep learning, Long Short Term Memory, Keypoints, MediaPipe, Asana, Real Time*

Table Of Contents

List of Figures

List of Tables

Abbreviations

API	Application programming interface
CNN	Convolution Neural Network
IMUs	Inertial Measurement Units
LSTM	Long Short Term Memory
RNN	Recurrent Neural Network
SDLC	Software Development Life Cycle

CHAPTER 1: INTRODUCTION

1.1 BACKGROUND THEORY

1.1.1 Introduction to Yoga

Yoga is derived from the Sanskrit word yuj, which means 'to connect.' It denotes 'unity' or 'oneness.' In spiritual words, this merger or joining is described as the union of the individual consciousness with the universal mind. On a more practical level, yoga is a method of relaxation, balancing and harmonizing the body, mind and emotions. This is done through the practice of asana, pranayama, mudra, bandha, shatkarma and meditation, and must be achieved before union can take place with the higher reality.[1].

The science of yoga begins to work on the outermost aspect of the personality, the physical body, which for most people is a practical and familiar starting point. Yoga aims to bring different bodily functions into perfect coordination so that they work for the good of the whole body. From the physical level yoga moves on to the mental and emotional levels. Yoga cannot prevent the cure of life but, it does present a proven method for coping with it.

Swami Sivananda Saraswati of Rishikesh explained yoga as an "Integration and harmony between thought, feeling and deed, or integration between head, heart and hand".He also said that, "Yoga is not an ancient myth buried in oblivion. It is the most valuable inheritance of the present. It is the essential need of today and the culture of tomorrow."

1.1.2 Introduction to Asana

Asana is a physical and mental state of being in which one can remain steady, peaceful, quiet, and pleasant. The Yoga Sutras of Patanjali have a short explanation of yogasanas: "Sthiram sukham aasanam," which translates to "pleasant and steady stance." So, we can see that yogasanas in this context are practised to develop the practitioner's ability to sit comfortably in one position for an extended length of time, as is necessary during meditation.

1.1.2.1 Bhujangasana

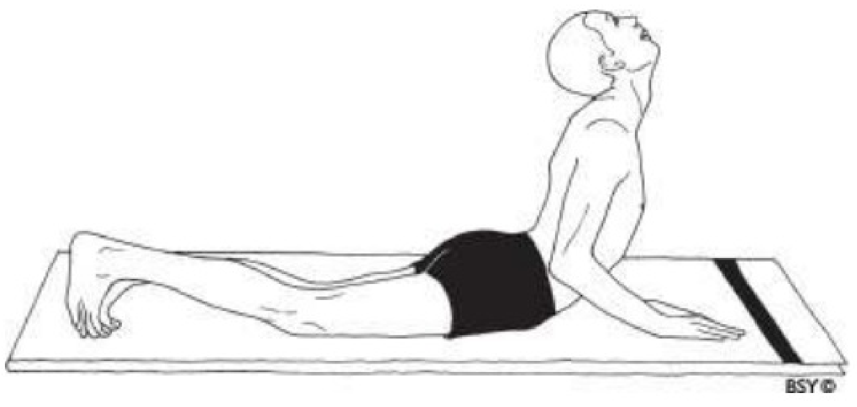


Figure 1.1: Bhujangasana

Bhujangasana or Cobra Pose is a yoga position where the hands and feet are kept in place while sliding the chest forward to raise the head, shoulders, and straighten the elbows to arch the back. The thighs and hips remain on the floor while the arms support the trunk, and the head is directed back with the gaze focused upward. Inhalation is done while raising the torso, and awareness should be focused on relaxation of the spine and Swadhisthana chakra. It is not recommended for individuals with peptic ulcer, hernia, intestinal tuberculosis, or hyperthyroidism.[1]

1.1.2.2 Shavasana

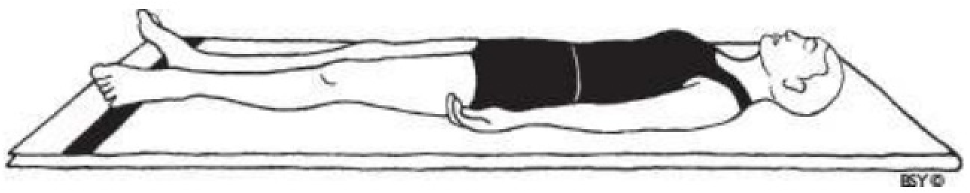


Figure 1.2: Shavasana

Shavasana, also known as Corpse Pose, is a yoga posture that is typically practiced at the end of a yoga class. It is a relaxation pose that aims to calm the body and mind, promoting deep rest and rejuvenation. Shavasana is an essential part of any yoga practice as it provides an opportunity to rest and integrate the benefits of the practice. It helps to reduce stress, anxiety, and tension in the body, and promotes better sleep and overall well-being. It is important to remember that Shavasana is a posture of complete surrender, so try to let go of any expectations or goals, and simply allow yourself to be present and fully relaxed.[1]

1.1.2.3 Padmasana

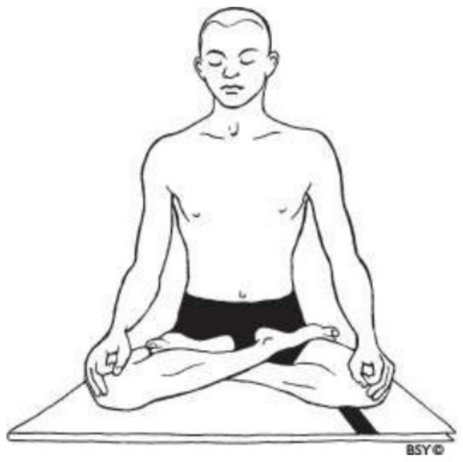


Figure 1.3: Padmasana

Padmasana, also known as lotus pose, is performed by sitting with the legs crossed and the feet placed on the opposite thighs. The hands are placed on the knees and the body is held upright. This posture helps in holding the body steady and calm, which is beneficial for meditation. It also directs the flow of prana from the perineum to the head, stimulates the digestive process, and reduces blood pressure. However, this pose should not be attempted by those with sciatica, weak or injured knees, or during pregnancy. It requires flexibility of the knees developed through practice of pre-meditation asanas.[1]

1.1.2.4 Tadasana

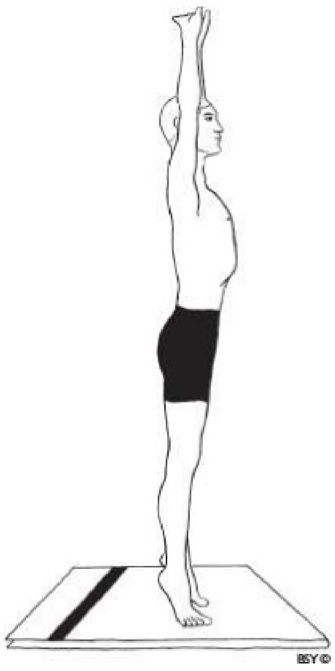


Figure 1.4: Tadasana

This is a set of instructions for a yoga practice consisting of 10 rounds. The practice involves standing with feet together or apart, raising the arms over the head, interlocking the fingers, placing the hands on top of the head, fixing the eyes on a point on the wall, inhaling and stretching the arms, shoulders, and chest upward while raising the heels, holding the position and breath for a few seconds, lowering the heels while exhaling and bringing the hands to the top of the head. It is recommended to relax for a few seconds between round.[1]

1.1.2.5 Vrikshasana

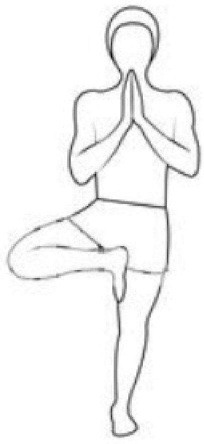


Figure 1.5: Vrikshasana

Vrikshasana, also known as Tree Pose, is a popular yoga posture that requires standing on one leg with the other leg bent and placed on the inner thigh of the standing leg, with the hands pressed together in front of the heart. This balancing pose is believed to help improve balance, stability, posture, and promote a sense of grounding and connection with the earth. To practice Vrikshasana, stand tall with your feet hip-width apart, shift your weight onto your left foot, and lift your right foot, placing it on the inner left thigh. Breathe deeply, focus your gaze, and hold the pose for several breaths before switching sides. By incorporating Vrikshasana into your yoga practice, practitioners can strengthen their body and mind and cultivate a greater sense of inner peace and well-being.[2]

1.1.2.6 Virabhadrasana 2

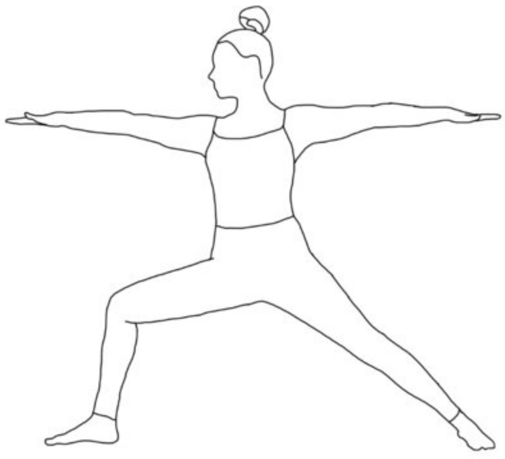


Figure 1.6: Virabhadrasana II

The pose starts with standing facing the long side of the mat with arms stretched straight out from the shoulders and feet parallel in a wide stance. Then, turn the right foot and knee to face the front of the mat and angle the left toes slightly in toward the upper left corner of the mat. Bend the right knee, stack it over the right ankle, and distribute the weight evenly between both legs. Stay here for 5-10 breaths, reaching strongly through both arms toward the front and back of the mat and turning the head to look past the right fingertips. To come out of the pose, exhale, press down through the feet, then inhale and straighten the legs, returning the feet to parallel facing the left long side of the mat.[1]

1.2 PROBLEM STATEMENT

Performing Yoga Asanas without proper knowledge can cause injury to the muscles and ligament.To perfect a Yoga Asana frequent and correct practice is necessary. Our system can classify among six Yoga Asanas and provide feedback incase there is a mistake in form, due to which chance of injury to muscles and ligament is decreased. Since our system can be used by a user alone frequent practice of Yoga Asanas is possible.

1.3 OBJECTIVES

- To detect Yoga Asana being performed by the user.
- To provide real time feedback on the Asana being performed by the user if the user is doing the asana incorrectly.

1.4 SCOPE AND APPLICATION

As of now there is no exact system that helps to perform yoga and provide proper feedback so our system has a great scope and if implemented properly, it could become helpful for the yoga trainer and practitioner. This system is useful for people that like to perform different yoga asanas on their own. This system decreases the chances of injuries while performing yoga. This system also helps in improving the posture of different asanas as it provides the real time feedback to the person. Our system's model can also be applicable for performing various sports and exercise prediction by providing the training.

CHAPTER 2: LITERATURE REVIEW

Yoga arose at the beginning of human civilization when humankind first realized their spiritual potential and began to evolve techniques to develop it. The yogic science was slowly developed by ancient sages all over the world. The essence of yoga has often been shrouded in or explained by different symbols, analogies and languages. Some traditions believe that yoga was a divine gift revealed to the ancient sages so that humankind could have the opportunity to realize its divine nature.

In the 21st century, beyond the needs of individuals, the underlying principles of yoga provide a real tool to combat social malaise. At a time when the world seems to be at a loss, rejecting past values without being able to establish new ones, yoga provides a means for people to find their own way of connecting with their true selves. Through this connection with their real selves, it is possible for people to manifest harmony in the current age, and for compassion to emerge where hitherto there has been none.

Human posture estimate has been accomplished using a variety of techniques. Robotics and computer engineering are two examples of applications where human activity recognition has been used. There has been a lot of effort put into establishing automated systems that assess yoga and sports like basketball [3] and cycling [4]. Many research have used different methodologies to classify yoga, such as posture estimation using the OpenPose algorithm [5] and posture identification using machine learning and deep learning techniques. In activity recognition or pose classification tasks too, there is a dependency between the previously performed action and the next action. In case of yoga as well, the context or information of initial or intermediary poses is important in predicting the final pose. Yoga can thus be thought of as a sequence of poses. This makes RNNs a suitable choice for yoga pose classification as sequential estimation of joint locations can be able to better capture the dependency between joint locations. For the same reason, C. Matthew et al. [6] use RNN for human pose estimation. K. Pothanaicker [7] proposed the integration of convolutional neural network and long short-term memory recurrent neural network for processing the video and achieved the accuracy of 90.05%.

2.1 EXISTING SYSTEM

Edwin W. Trejo and Peijiang Yuan [8] proposed an interactive system capable of recognizing 6 poses for learning Yoga that can track up to 6 people at a time that used Adaboost algorithm and achieved 94.8% accuracy. L. Chaihouoy et al. [9] proposed a transfer learning based yoga posture coaching system which had an accuracy of 98.43%

Patil et al. [10] suggested a 'Yoga Tutor' project that uses speeded up robust characteristics to determine the difference in postures between a practitioner and an expert (SURF). However, using merely the contour information to compare and describe the postures is insufficient. Luo et al. [11] suggested a Yoga training system that includes an interface suit with 16 inertial measurement units (IMUs) and six tactors that is unobtrusive to the user and can alter the user's ability to do asana in a natural manner.

A. Chaudhari et al. [12] proposed a deep learning model for building a system that uses convolution neural networks (CNN) for yoga pose identification and a human joints localization model, followed by a mechanism for identifying posture mistakes and achieved an classification accuracy of 95% for pose identification. Kothari S. [13] proposed a yoga pose classification using OpenPose, PoseNet, PIFPAF and CNN and achieved an accuracy of 98.58%.

The work proposed in [14] uses OpenPose for initial keypoint identification followed by CNN for classification of yoga poses. However, they achieve an accuracy of only 78% which could be due to the limited dataset they used or architecture and hyperparameter tuning of their CNN model.

2.2 LIMITATIONS OF EXISTING SYSTEM

Author in [8] has proposed a model that uses Microsoft kinect sensor for motion detection Kinet sensor is costlier than webcam so the implementation of this sensor increases the cost of the system. In [9], author uses image for training asanas due to which the earlier steps are not considered while performing the asana. In [10], author has proposed a model that uses SURF algorithm which makes it unstable to rotation and it doesn't work properly in poor illumination condition. Interface suit is used in [11] and done in virtual environment which is very expensive and not feasible for the consumer.

2.3 SOLUTION PROPOSED BY OUR SYSTEM

- Our system has used the combination of MediaPipe pose model and LSTM to predict the asanas. This model was trained by providing video input due to which it considers entire movement while performing an asana which is not possible in a image trained model.
- Our system has used the webcam of the laptop and mobile phone for video input so the cost of implementing the system is cheaper than using Microsoft Kinect or Interface Suit.
- Our system classifies and provides correction for six different Yoga asanas.

CHAPTER 3: METHODOLOGY

3.1 PROCESS MODEL

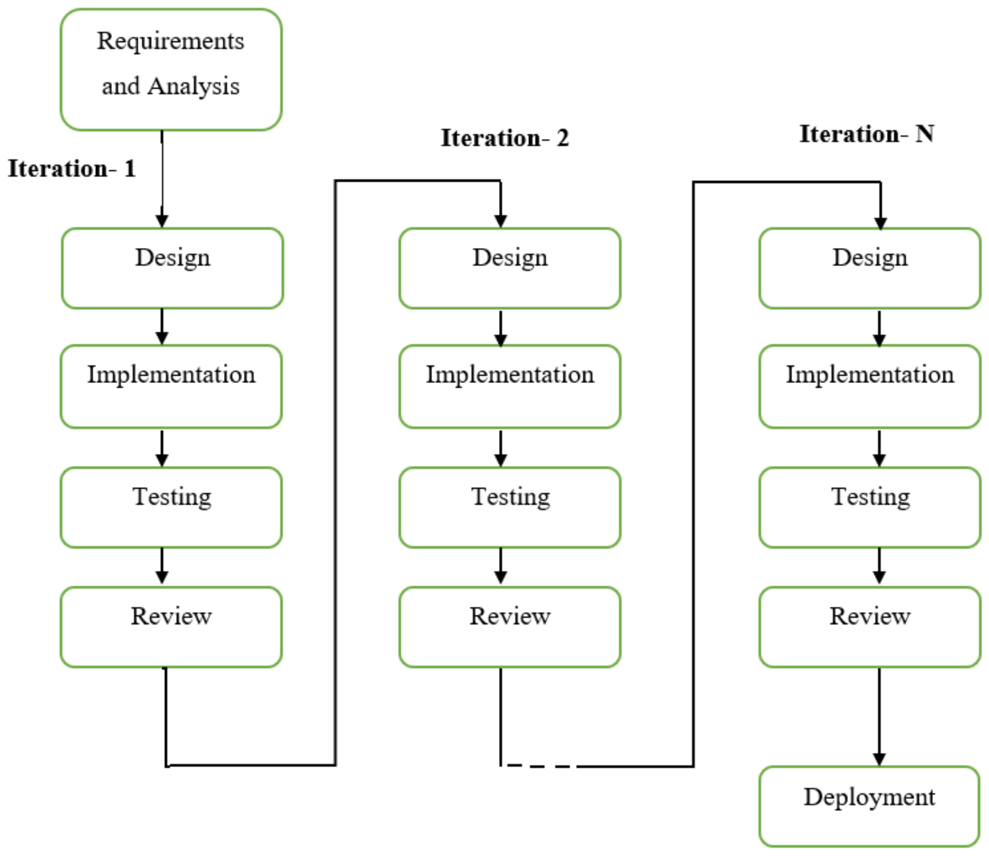


Figure 3.1: Iterative Process Model

In this Model, we can start with some of the software specifications and develop the first version of the software. After the first version if there is a need to change the software, then an updated version of the software is created with a new iteration. Every release of the Iterative Model finishes in an exact and fixed period that is called iteration.The Iterative Model allows the accessing earlier phases, in which the variations made, respectively. The final output of the project renewed at the end of the Software Development Life Cycle (SDLC) process.

1. **Requirement gathering analysis:** In this phase, requirements were gathered by collecting the necessary information from the yoga guru. Yoga videos for input

were collected and were analysed by yoga guru. After all of this, we skipped to the next phase.

2. **Design:** In the design phase, we designed the software using different diagrams like activity diagram, Sequence Diagram , Use Case Diagram etc.
3. **Implementation:** In the implementation, requirements were written in the coding language and transformed into computer programs which gave a final product as a yoga asana classifying model and a feedback generator .
4. **Testing:** After completing the coding phase, we performed tesing on different yoga asana at different location
5. **Review:** Review phase was performed to check the behavior and validity of the developed product.
6. **Deployment** After completing all the phases, we deployed the system to its work environment.

3.2 BLOCK DIAGRAM

3.2.1 Asana Classification

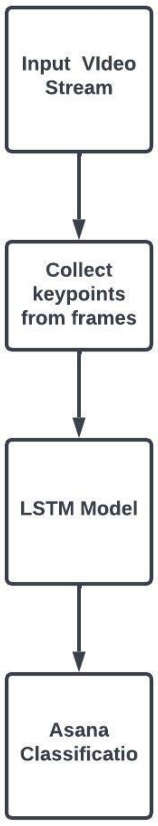


Figure 3.2: Asana Classification

1. **Input Video Stream**
   OpenCV has been used to take the video input from the user for the classification. The device used by the user can be webcam or mobile phone. We have used mobile phone's camera to take the input video.

2. **Extract Keypoints**
   After the video of the Asana has been taken from the mobile phone, it is passed to the Mediapipe Pose model that has been used to extract Keypoints i.e. Pose Landmarks from 20 frames of the video. 33 Keypoints are extracted from each frames. Each keypoint consists of a X-axis, Y-axis and Z-axis value along with visibility of the keypoint.

3. **LSTM Model**

   The input of shape (20,132) is passed to a LSTM model where 20 is the number of frames and 132 is the flattened list of (33,4) matrix where 33 is the no of keypoints and 4 is the X-axis, Y-axis, Z axis value with visibility of the keypoints.

4. **Pose Classification**

   The LSTM model provide classifciation of the performed yoga pose by keeping track of each frames.

.

3.2.2 Feedback Generation

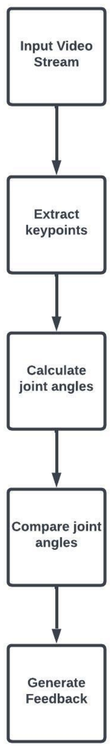


Figure 3.3: Feedback Generation

1. **Input Video Stream**

   OpenCV has been used to open the video input device which can be a webcam or a camera from any external device.

2. **Extract Keypoints**

   The Keypoints or Landmark of every frame has been extracted for Feedback Generation, 33 keypoints are extracted and only X-axis and Y-axis value is used.

3. **Calculate joint angle**

   The angle of the joint is calculated in this step

4. **Compare joint angle**

   The calculated joint angle values are compared with the ideal value of joint angle of the respective pose. This has been done for each frame

5. **Generate Feedback**

   Feedback has been generated as per the result that is obtained from comparing the joint angle

3.3 DATA COLLECTION

The dataset consists of videos of six Yoga Asana as show below:

1. Bhujangasana
2. Shavasana
3. Padmasana
4. Tadasana
5. Vrikshasana
6. Virabhadrasana II

The dataset was collected using a phone camera at 30 fps 1080p at variable distances of about 4 - 6m for various Asana with 5 different people performing the asana. It was collected under the supervision of Mr Shanta Kharel, Yoga Instructor in KEC. The dataset were collected in the yoga hall of Kathmandu Engineering College. There is a split of 80-20 between Training and Validation.

3.3.1 Dataset Summary

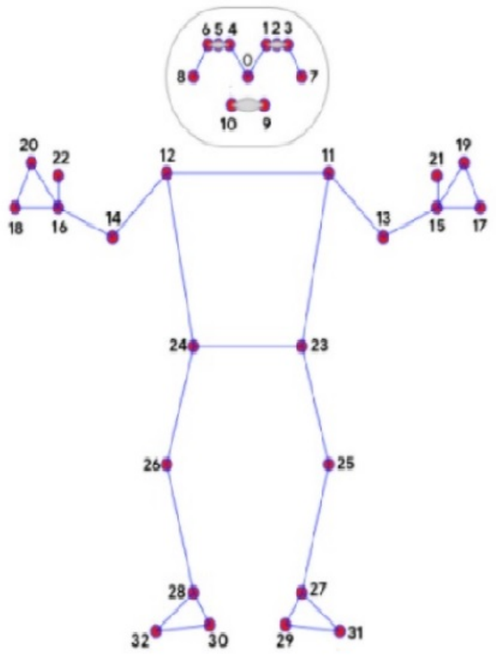
The total number of videos collected for each Yoga Asanas is listed in the table below :

Asana	**Number of Videos**
Bhujangasana	24
Shavasana	20
Padmasana	24
Tadasana	24
Vrikshasana	21
Virabhadrasana II	19

Table 3.1: Dataset Summary for Six different Yoga Asanas

3.4 ALGORITHM

3.4.1 MediaPipe Pose Model



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

Figure 3.4: Keypoints extracted by Mediapipe pose model

MediaPipe Pose is a ML solution for high-fidelity body pose tracking, inferring 33 3D landmarks and background segmentation mask on the whole body from RGB video frames utilizing Mediapipe's BlazePose research that also powers the ML Kit Pose Detection API. Current state-of-the-art approaches rely primarily on powerful desktop environments for inference, whereas this method achieves real-time performance on most modern mobile phones, desktops/laptops, in python and even on the web. The detector is inspired by MediaPipe's own lightweight BlazeFace model, used in MediaPipe Face Detection, as a proxy for a person detector. It explicitly predicts two additional virtual key points that firmly describe the human body center, rotation and scale as a circle. Inspired by Leonardo's Vitruvian man, It predicts the midpoint of a person's hips, the radius of a circle circumscribing the whole person, and the incline angle of the line connecting the shoulder and hip midpoints.

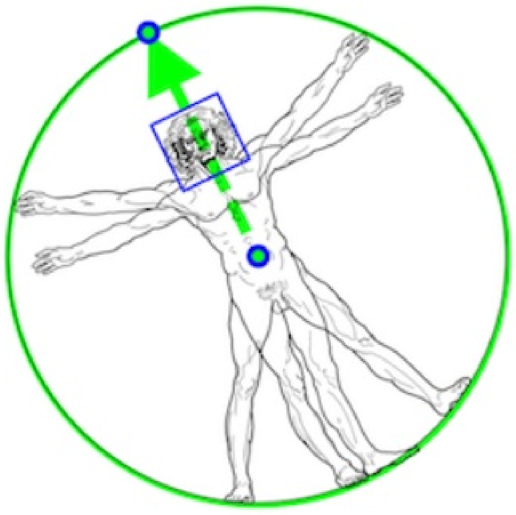


Figure 3.5: Virtuvian Man

The detector is based on a lightweight model called BlazeFace, originally developed for face detection, which has been modified to detect people by predicting key features such as the center of the body, rotation, and scale as a circle. The model also predicts the midpoint of a person's hips, the radius of a circle encompassing the entire person, and the angle of the line connecting the midpoint of the shoulders and hips, inspired by Leonardo da Vinci's Vitruvian man.

We have used MediaPipe Pose Model in both Asana Classification and Feedback Generation. In Asana Classification Mediapipe Pose Model is used to extract key points from 20 frames of the video and the key points are then fed to the LSTM which makes the necessary prediction. In Feedback Generation key points is extracted from every frame and key points are used to calculate joint angle.[15]

3.4.2 Long-Short Term Memory (LSTM)

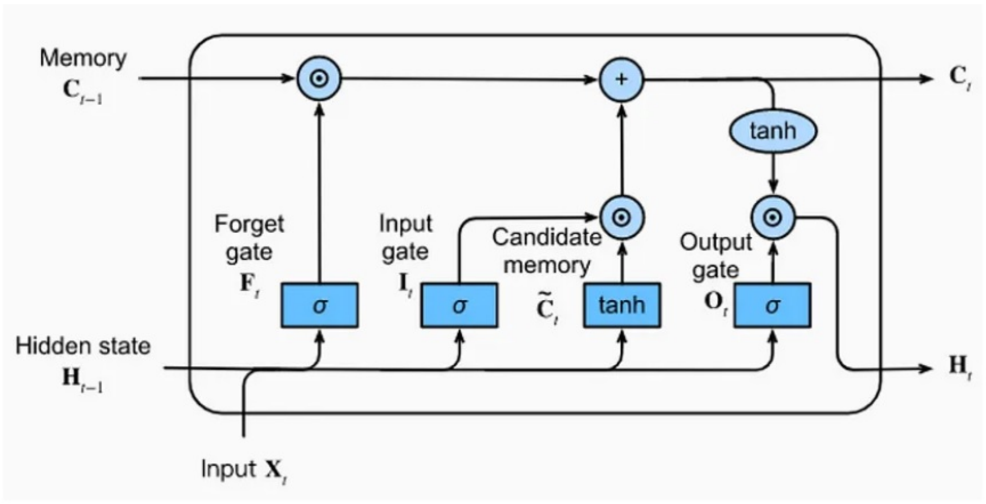


Figure 3.6: Long Short Term Memory Neural Network

LSTM stands for long short-term memory networks, used in the field of Deep Learning. It is a variety of recurrent neural networks(RNNs) that are capable of learning long-term dependencies, especially in sequence prediction problems. LSTM has feedback connections, i.e., it is capable of processing the entire sequence of data, apart from single data points such as images. This finds application in speech recognition, machine translation, etc. LSTM is a special kind of RNN, which shows outstanding performance on a large variety of problems.

The central role of an LSTM model is held by a memory cell known as a 'cell state' that maintains its state over time. The cell state is the horizontal line that runs through the bottom of the diagram above. Information can be added to or removed from the cell state in LSTM and is regulated by gates. These gates optionally let the information flow in and out of the cell. It contains a point wise multiplication operation and a sigmoid neural net layer that assist the mechanism. The sigmoid layer gives out numbers between zero and one, where zero means 'nothing should be let through,' and one means 'everything should be let through.[16]

For Forget gate,

$$F_t = \sigma(X_t * U_f + H_t - 1 * W_f)......................................(1)$$

For Candidate memory,

$$\bar{C_t} = tanh(X_t * U_c + h_{t-1} * W_c)................................(2)$$

For Input Gate,

$$I_t = \sigma(X_t * U_i + H_{t-1} * W_i)....................................(3)$$

For Output Gate,

$$O_t = \sigma(X_t * U_o + H_{t-1} * W_o).....................................(4)$$

For Memory Cell,

$$C_t = f_t * C_{t-1} + I_t * \bar{C_t}...(5)$$

For Current Cell Output

$$H_t = O_t * tanh(C_t)...(6)$$

Here,

W, U = weight vectors for forget gate (f) candidate (c) , i/p gate(I) and o/p gate (O) candidate (c), i/p gate (i) and o/p gate(O).

The system consists of a deep learning LSTM model built using Keras API. The model consists of 6 layers in a sequential order. The model takes as input a sequence of 20 time steps, where each time step represents a frame in a video and each frame contains a vector of 132 features. The output of the model is a probability distribution over 6 possible classes, which correspond to different asanas that may be happening in the video. The first layer is an LSTM layer with 64 memory units, which takes input with the shape of (20,132) and returns the output at each time step, instead of just output at the final step and uses the rectified linear unit (relu) activation function. The second layer is a dropout layer which randomly drops out 20% of the units in the previous layer. Dropout is the regularization technique that helps prevent over fitting by forcing the network to learn more robust features. The third layer is another LSTM layer with 128 memory units, returning output at each time step and using the relu activation function. The fourth layer is another dropout layer which drops out 20% of the units. The fifth layer is final LSTM layer with 64 units and only returns the output at the final step and uses the relu activation function. The sixth layer is a fully connected Dense layer with 64 units and ReLU activation. This layer takes the output of the previous LSTM layer and applies a linear transformation to it. The output of this layer is a vector with 64 elements. The seventh layer is another dropout layer, which drops out 20% of the units. The eighth layer is another fully connected layer with 32 units and ReLU activation. This layer further reduces the dimensionality of the data and extracts higher-level features. The ninth and final layer is a fully connected layer with 6 units and softmax activation. This layer produces the output probabilities for each of the 6 possible classes. Softmax is used as the activation function because it ensures that the output probabilities sum to 1.0. Adam Optimizer was used in training our model with the learning rate of 0.0001. This optimization algorithm was used to update the weights of the neural network during training. It adapts the learning rate based on the gradients of the parameters, allowing it to converge more quickly and efficiently. Categorical Crossentropy was used to calculate the loss and accuracy was used to evaluate the performance of the model. The model was trained for 200 epoch with a batch size of 4.

3.4.3 Joint Angle Classification

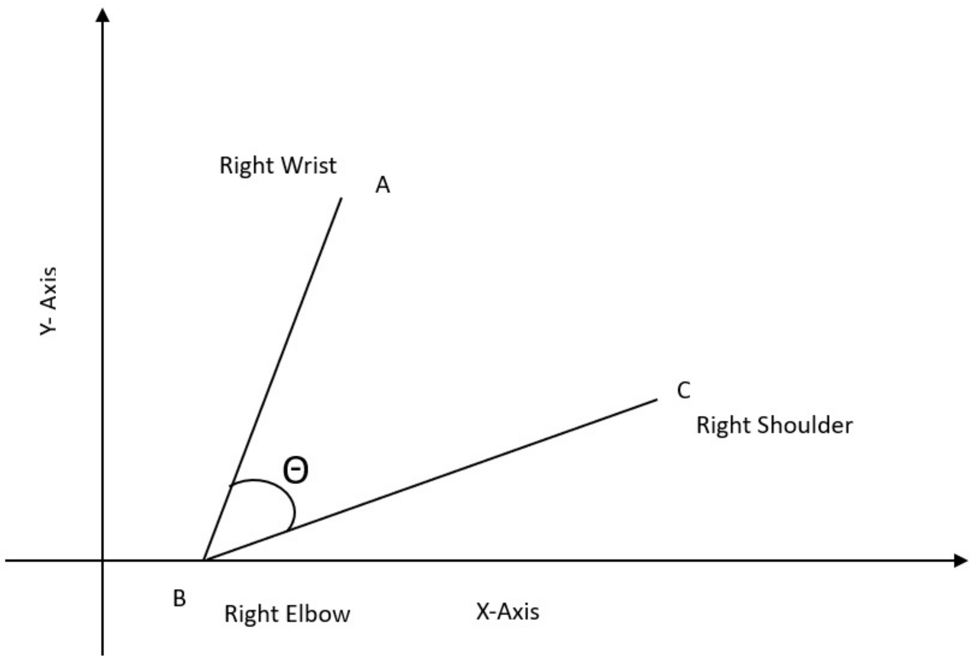


Figure 3.7: Angle Calculation Representation of Right Elbow

The system calculates the angle between the joints of a person's body, given the 2D coordinates (x and y) of the landmarks obtained from the Mediapipe Pose model. Specifically, it finds the angle between three points A, B, and C, where B is the joint whose angle is being calculated, A is the landmark of the joint connected to B, and C is the landmark of the joint on the other side of B.The formula given below is used to calculate the angle:

$$\theta = \tan^{-1}(\text{Slope of AB}) - \tan^{-1}(\text{Slope of BC})\text{.................}(7)$$

For example, for the calculation of the angle of the elbow joint, we passed the coordinates of the shoulder landmark as C, the coordinates of the elbow landmark as B, and the coordinates of the wrist landmark as A. The function converts the coordinates into NumPy arrays and uses the arctan2 function to calculate the angle in radians. Then, it converts the angle to degrees and returns the absolute value of the angle (since the arctan2 function can return negative values). If the angle is greater than 180 degrees, the function subtracts it from 360 to get the angle of the joints .

3.5 NECESSARY UML DIAGRAMS

3.5.1 Use Case Diagram

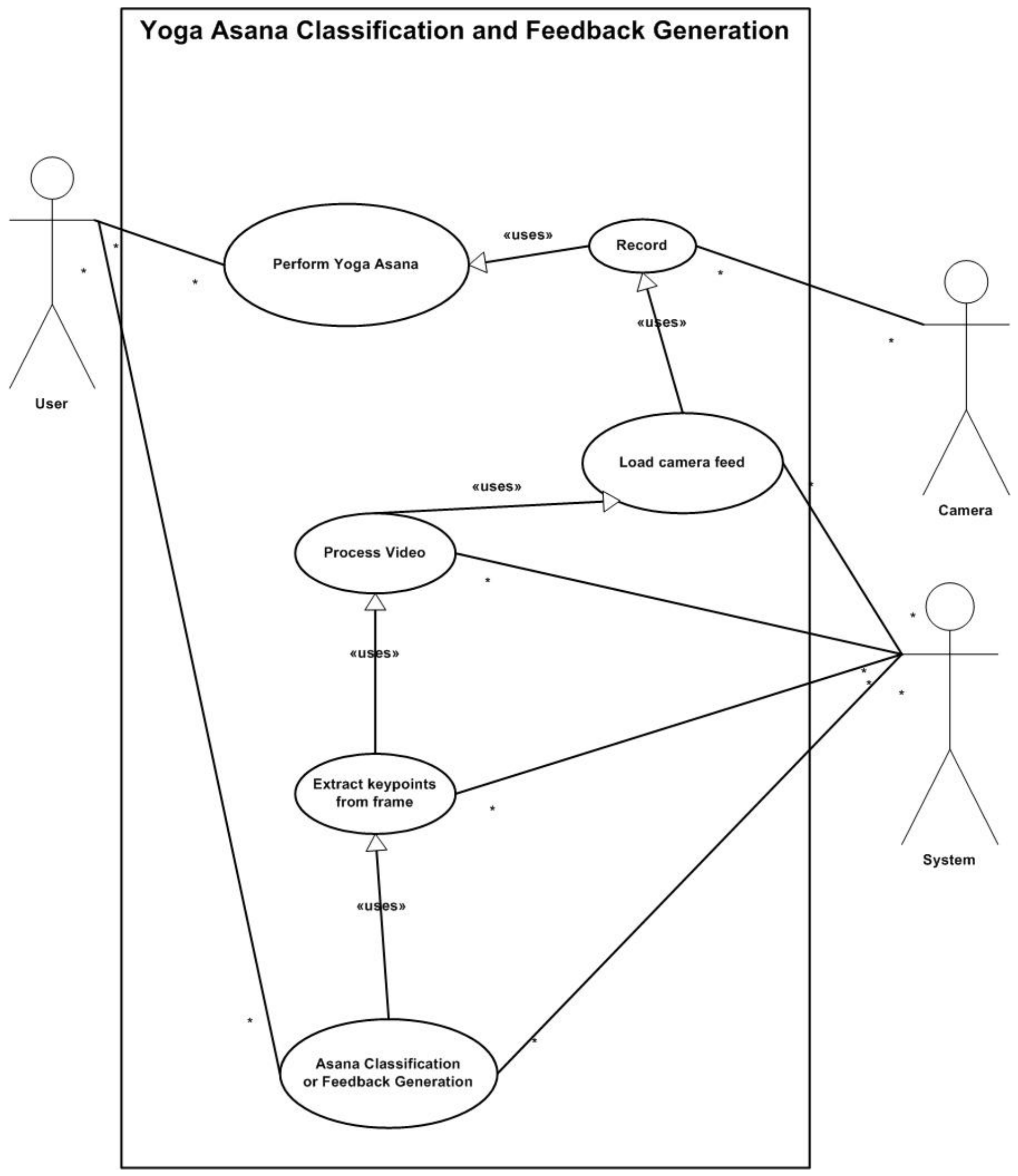

Yoga Asana Classification and Feedback Generation
User
Camera
System
Perform Yoga Asana
Record
«uses»
Load camera feed
Process Video
Extract keypoints from frame
Asana Classification or Feedback Generation

Figure 3.8: Use Case Diagram of Yoga asana classification and feedback generation

There are three actor in the system: User, Camera and System. Camera records the asana performed by the user. System then uses the video recorded by the camera and then process the video. The video must be loaded in order to process it. After the processing of the video the system extracts the keypoints from each frame of the video and finally Asana classification or feedback generation case classifies the pose or generates the feedback of the desired pose using the keypoints from the frame..

3.5.2 Activity Diagram

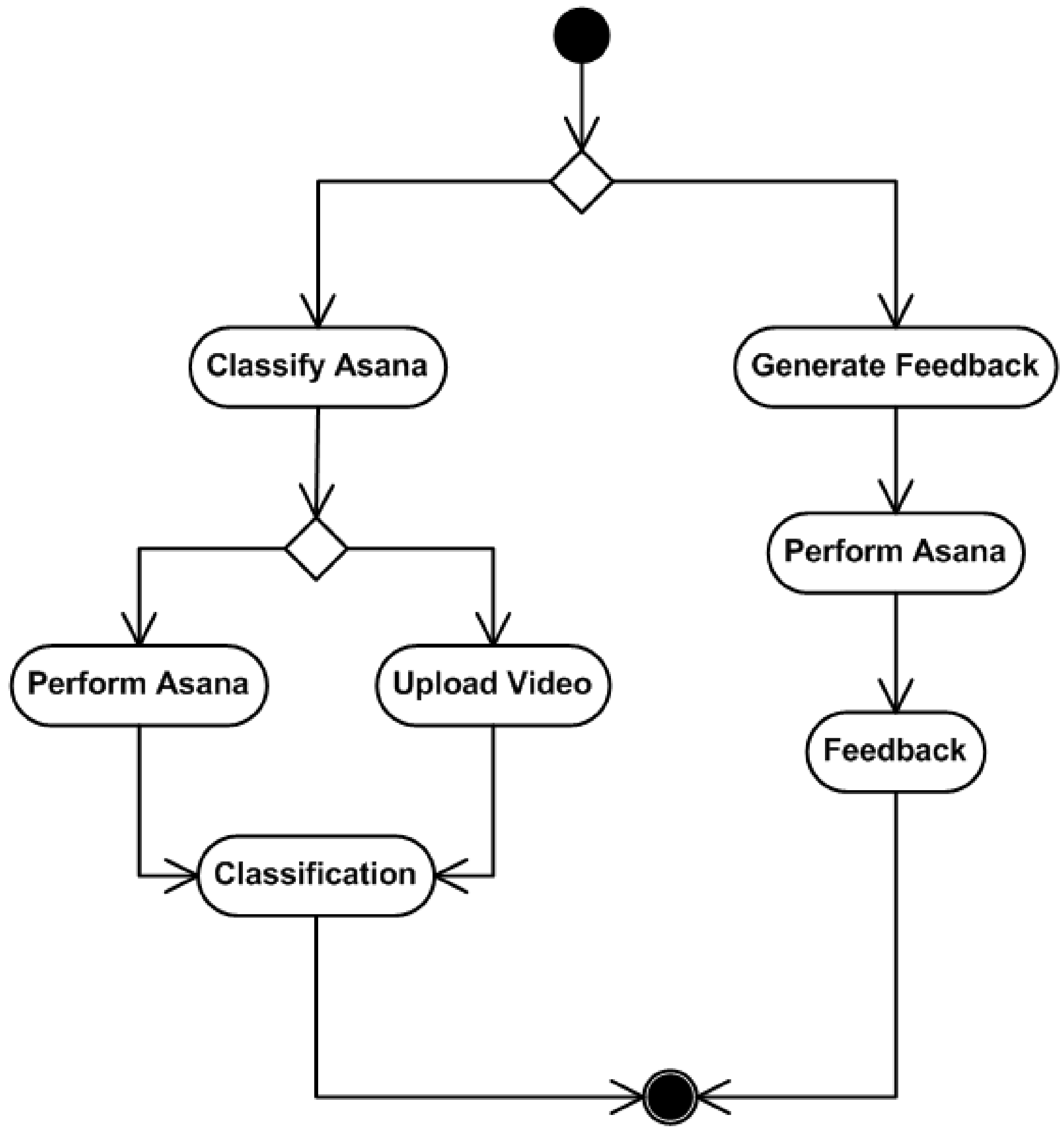


Figure 3.9: Activity Diagram of yoga asana classification and feedback generation

The activity diagram explains the overall activity flow of our project. Initially a decision need to be made where the user chooses to either classify the asana or generate the feedback of the asana. If the user chooses to classify the asana, they need to either perform the asana through live video camera or upload the available yoga asana videos in order to classify the asana. If the user chooses to generate feedback, they need to perform the asana in front of the camera and the real time feedback for correcting the asana is provided.

3.5.3 Sequence Diagram

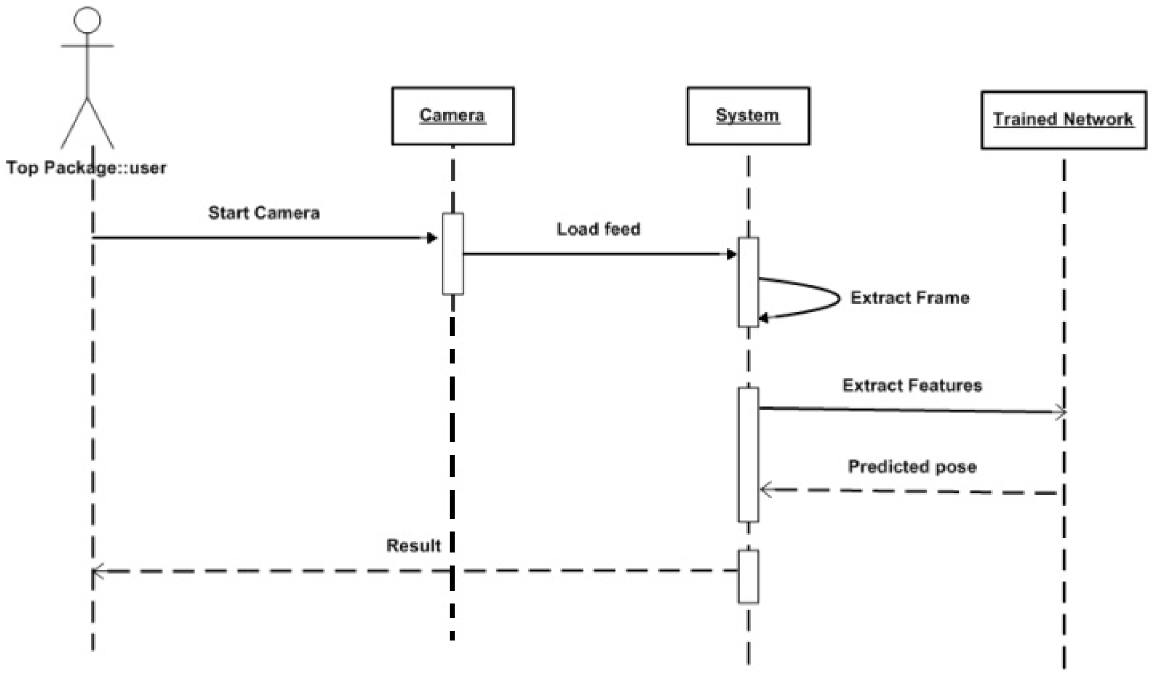


Figure 3.10: Sequence Diagram of yoga asana classification and feedback generation

The sequence diagram above shows an actor which is represented by the term “user” and has three objects, namely “camera”, “system” and “trained network”. The process begins when the user sends a “start camera” message to the camera object, indicating that they want to start capturing video data. Upon receiving the start camera message, the camera object sends a load feed message to the system object, instructing it to load the video feed. The system object then sends an extract frame message to itself, which initiates the process of extracting individual frames from the video feed. After the system object extracts a frame, it sends and extract features message to trained network object which is responsible for analysing the frame and extracting the relevant features which contains the keypoints in the frame. The trained network then predicts the asana based on the features and predicts the asana. The final result is the provided to the user.

3.6 TOOLS USED

3.6.1 Python

Various python libraries has been used for the completion of out project. Numpy has been used for matrix multiplication and joint angle calculation. Opencv library has been used for the access of the webcam and deal with the processing of the video. OS library has been used to preform operating system dependent functionality. Matplotlib and seaborn library has been used for plotting the confusion matrix.

3.6.2 Tensorflow and Keras

Tensorflow and Keras has been used for preparing data for training and then Keras to build our LSTM model.It was used to train the model and perform prediction on the data. At the end performance measures was also calculated using Keras.

3.6.3 Flask

We have use Flask for representing our project in web. Flask has been used to compile the Html, Css, Javascript and the machine learning model.

3.6.4 HTML, CSS and JavaScript

Html, Css and Javascript has been used to create the frontend layout for the representation of the project in web.

3.6.5 TensorBoard

TensorBoard has been used to create the graph of accuracy and loss using the log files saved by the tensorflow platform during the training of the model.

3.7 VERIFICATION AND VALIDATION

3.7.1 Model Perfomance for Training and Testing Data

3.7.1.1 Training Performance

Epoch : 200

Training accuracy : 1

Training loss : 6.3726e-05

Validation accuracy: 1

Validation loss: 0.0012

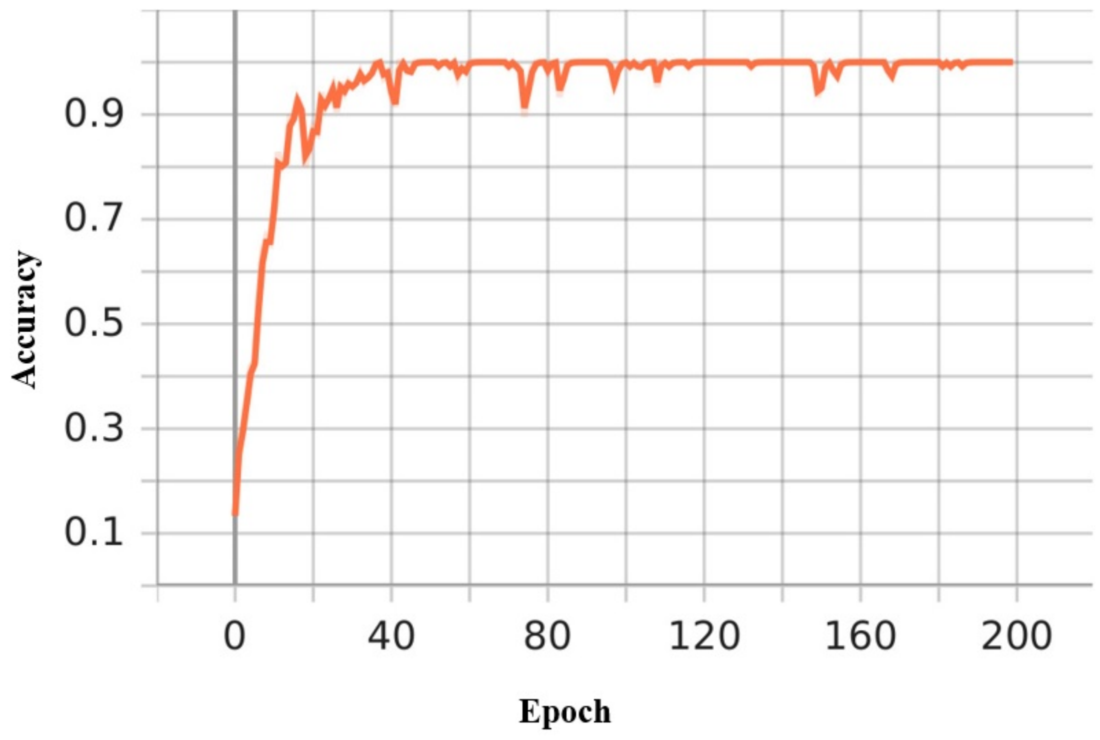


Figure 3.11: Training Accuracy vs Epochs curve

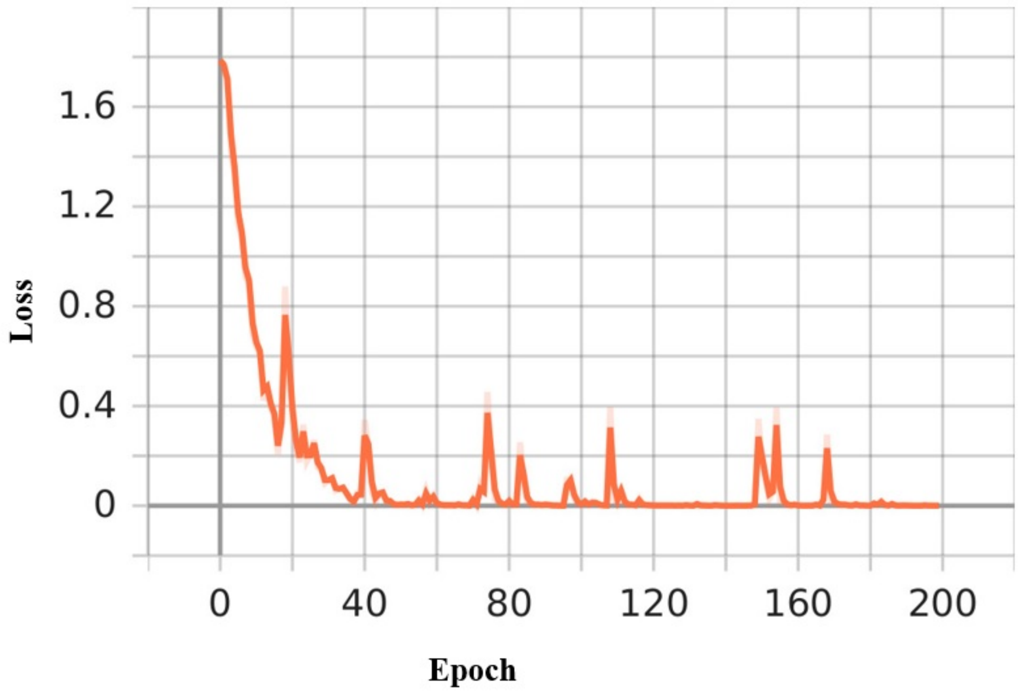


Figure 3.12: Training loss vs Epochs curve

3.7.1.2 MODEL PERFORMANCE MATRIX

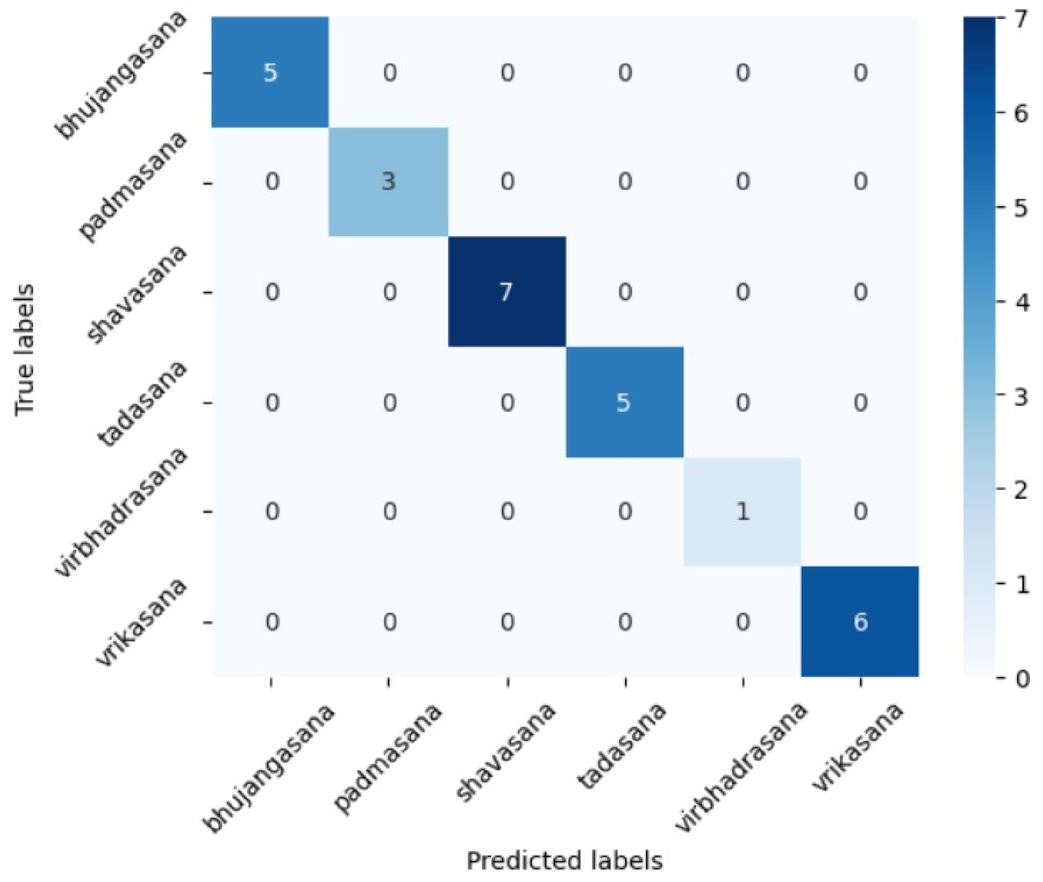


Figure 3.13: Confusion Matrix of the validation set

The validation accuracy of the model is 1.00. Various performance measures calculated from the above confusion matrix given in the table below :

Asana	Precision	Recall
Bhujangasana	1.00	1.00
Shavasana	1.00	1.00
Padmasana	1.00	1.00
Tadasana	1.00	1.00
Virbhadrasana	1.00	1.00
Vrikasana	1.00	1.00

Table 3.2: Performance Measures for the validation dataset

CHAPTER 4: EPILOGUE

4.1 RESULT

Hence, we finally compiled a program that recognizes the yoga pose performed by the user and also provide the feedback of the pose to the user. Each pose performed by the user goes through the series of processes such as frames extraction from the video, keypoint extraction using mediapipe, pose classification by the help of LSTM model. The feedback for the pose is generated by calculating the angle between the joints.

4.2 CONCLUSION

In conclusion we were able to obtain a system that is capable of predicting among the six Yoga Asana performed by the user and the system could also provide feedback to the user performing Asana. The Asana classifier model is a combination of Mediapipe Pose model with LSTM. The model is trained for 200 epochs and has been optimized using the Adams Optimizer algorithm with a batch size of 4. The learning rate of the optimizer used was 0.0001 and the loss function used was categorical crossentropy. The model optimizes for the loss function that is the categorical cross-entropy. For all six classes of yoga asana the precision and recall is 1. The system can be used by people to practise and learn yoga asanas.

4.3 FUTURE ENHANCEMENT

The current system has the ability to perform two tasks: Yoga Asana Classification and providing feedback while performing Yoga asanas. However, these tasks are performed separately. In the future theses tasks can be combined, in order to combine these functionalities for an enhanced experience, the Yoga Asana Classifier must achieve 100% accuracy. To achieve this, a large amount of data on various Yoga Asanas is required. The current classifier is capable of classifying 6 Yoga Asanas, but this can be expanded to classify all 84 Yoga Asanas.

REFERENCES

[1] S. Satyananda, *Asana pranayama mudra bandha.* Bihar: Yoga Publications Trust, 1996.

[2] [Online]. Available: https://www.nepalyogaacademy.com/poses/tree-pose-vrikshasana/

[3] P.-F. Pai, L.-H. ChangLiao, and K.-P. Lin, “Analyzing basketball games by a support vector machines with decision tree model,” *Neural Computing and Applications*, vol. 28, no. 12, pp. 4159–4167, 2017.

[4] S. Haque, A. Rabby, M. A. Laboni, N. Neehal, and S. A. Hossain, “Exnet: deep neural network for exercise pose detection,” in *International Conference on Recent Trends in Image Processing and Pattern Recognition.* Springer, 2018, pp. 186–193.

[5] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, “Realtime multi-person 2d pose estimation using part affinity fields,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7291–7299.

[6] M. Chen and M. Low, “Recurrent human pose estimation.”

[7] K. Pothanaicker, “Human action recognition using cnn and lstm-rnn with attention model,” *Intl Journal of Innovative Tech. and Exploring Eng*, vol. 8, no. 8, 2019.

[8] E. W. Trejo and P. Yuan, “Recognition of yoga poses through an interactive system with kinect device,” in *2018 2nd International Conference on Robotics and Automation Sciences (ICRAS).* IEEE, 2018, pp. 1–5.

[9] C. Long, E. Jo, and Y. Nam, “Development of a yoga posture coaching system using an interactive display based on transfer learning,” *The Journal of Supercomputing*, vol. 78, no. 4, pp. 5269–5284, 2022.

[10] S. Patil, A. Pawar, A. Peshave, A. N. Ansari, and A. Navada, “Yoga tutor visualization and analysis using surf algorithm,” in *2011 IEEE control and system graduate research colloquium.* IEEE, 2011, pp. 43–46.

[11] Z. Luo, W. Yang, Z. Q. Ding, L. Liu, I.-M. Chen, S. H. Yeo, K. V. Ling, and H. B.-L. Duh, "“left arm up!” interactive yoga training in virtual environment," in *2011 IEEE virtual reality conference*. IEEE, 2011, pp. 261–262.

[12] A. Chaudhari, O. Dalvi, O. Ramade, and D. Ambawade, “Yog-guru: Real-time yoga pose correction system using deep learning methods,” in *2021 International Conference on Communication information and Computing Technology (ICCICT)*. IEEE, 2021, pp. 1–6.

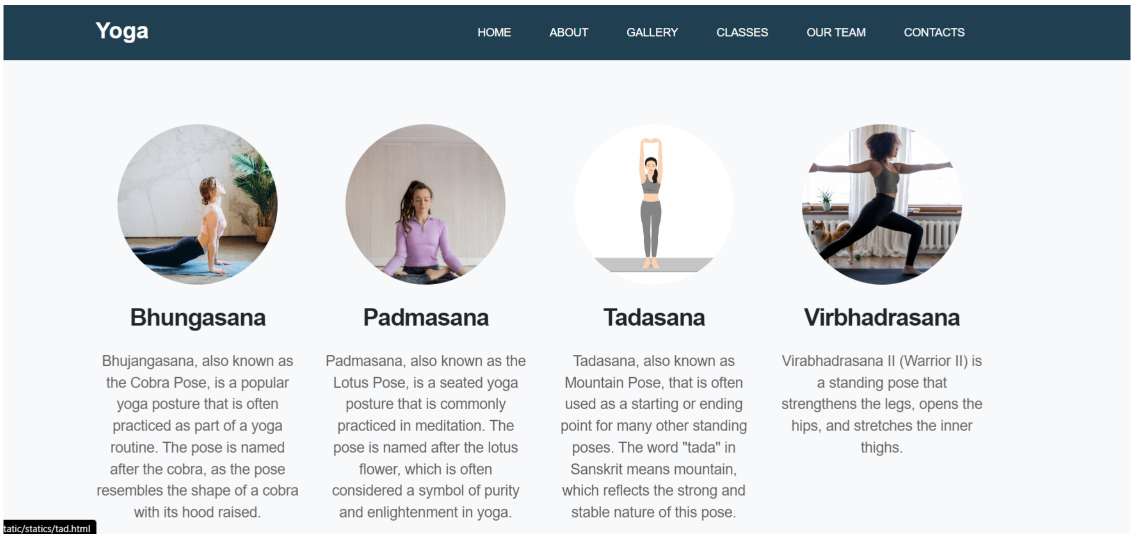
[13] S. Kothari, “Yoga pose classification using deep learning,” 2020.

[14] A. Lai, B. Reddy, and B. Vlijmen, “Yog. ai: deep learning for yoga,” 2019.

[15] [Online]. Available: https://google.github.io/mediapipe/solutions/pose.html

[16] “What is lstm - introduction to long short term memory,” Feb 2022. [Online]. Available: https://intellipaat.com/blog/what-is-lstm/

SCREENSHOTS



Bhungasana

Bhujangasana, also known as the Cobra Pose, is a popular yoga posture that is often practiced as part of a yoga routine. The pose is named after the cobra, as the pose resembles the shape of a cobra with its hood raised.

Padmasana

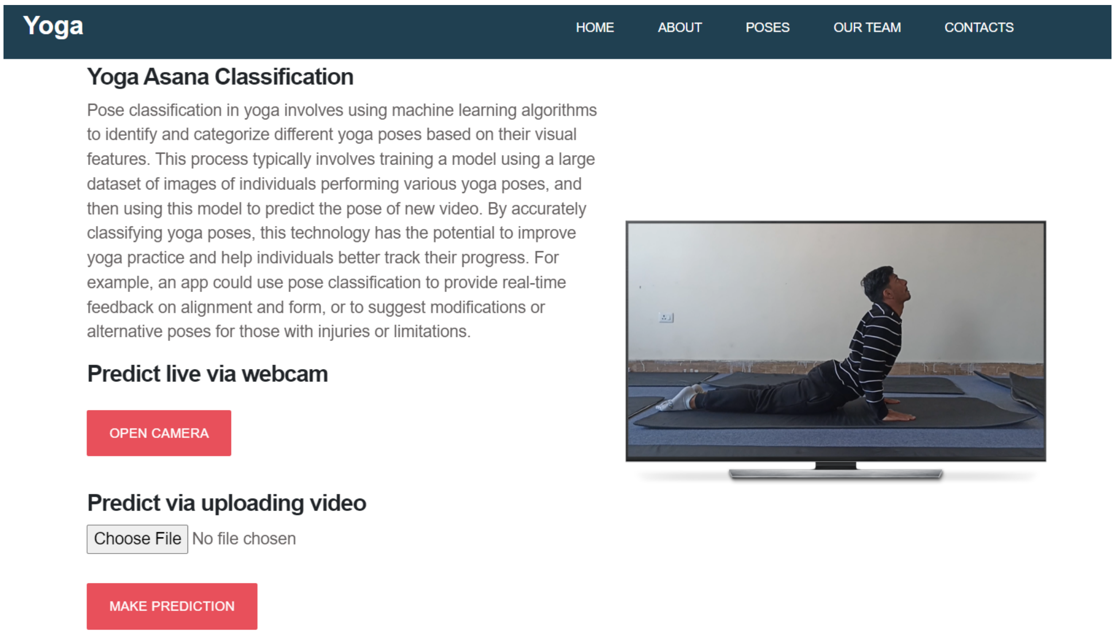
Padmasana, also known as the Lotus Pose, is a seated yoga posture that is commonly practiced in meditation. The pose is named after the lotus flower, which is often considered a symbol of purity and enlightenment in yoga.

Tadasana

Tadasana, also known as Mountain Pose, that is often used as a starting or ending point for many other standing poses. The word "tada" in Sanskrit means mountain, which reflects the strong and stable nature of this pose.

Virbhadrasana

Virabhadrasana II (Warrior II) is a standing pose that strengthens the legs, opens the hips, and stretches the inner thighs.

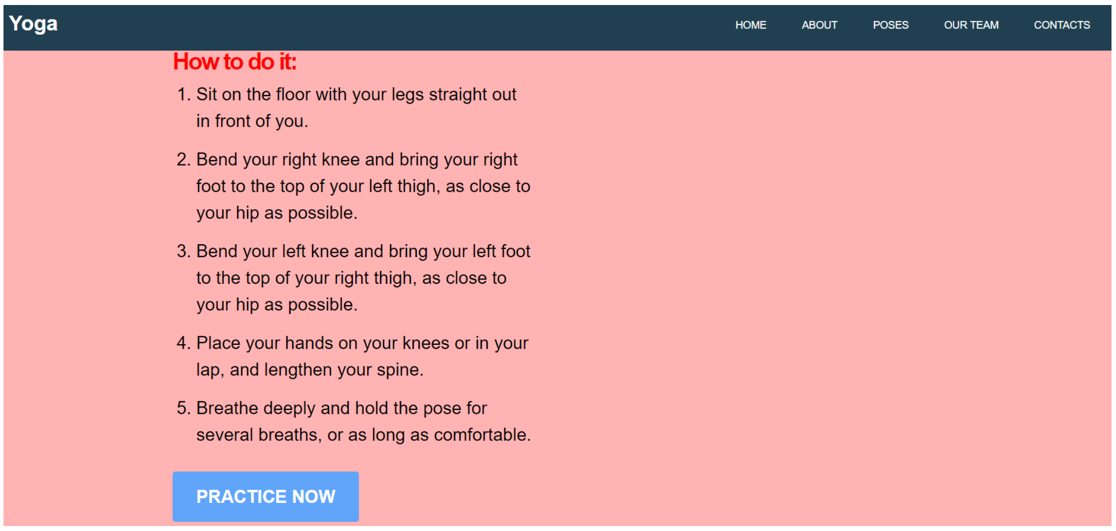


Yoga Asana Classification

Pose classification in yoga involves using machine learning algorithms to identify and categorize different yoga poses based on their visual features. This process typically involves training a model using a large dataset of images of individuals performing various yoga poses, and then using this model to predict the pose of new video. By accurately classifying yoga poses, this technology has the potential to improve yoga practice and help individuals better track their progress. For example, an app could use pose classification to provide real-time feedback on alignment and form, or to suggest modifications or alternative poses for those with injuries or limitations.

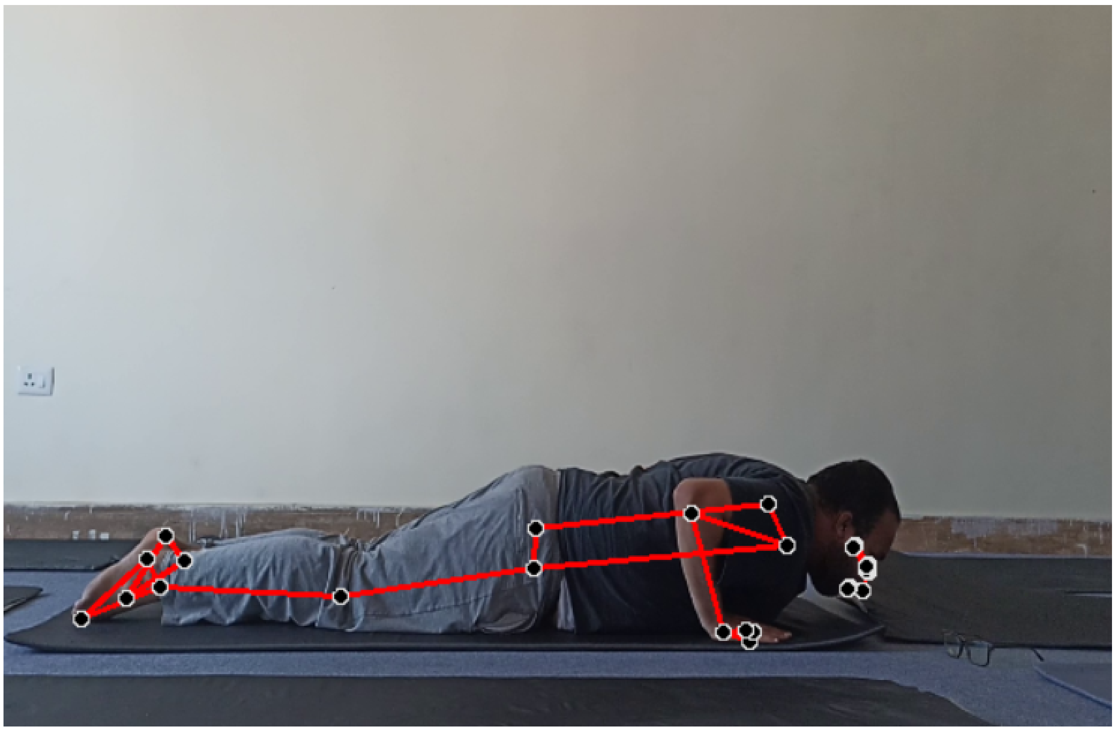
Predict live via webcam

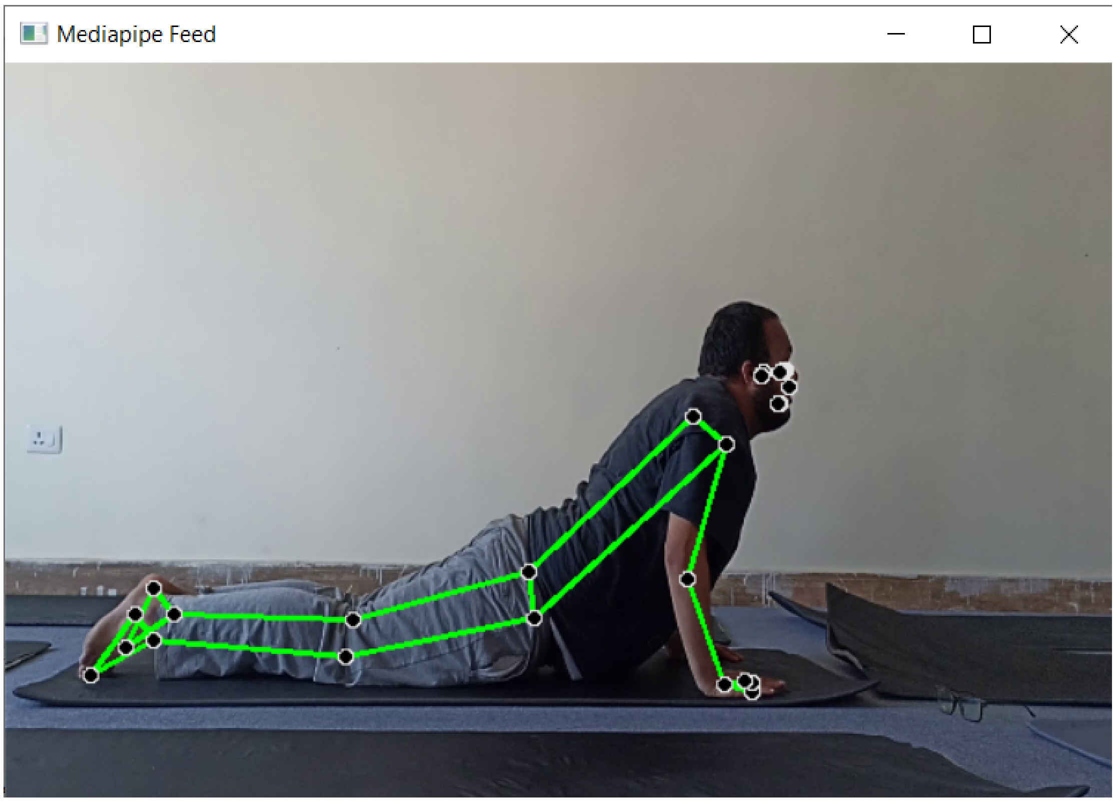
Predict via uploading video



How to do it:

1. Sit on the floor with your legs straight out in front of you.
2. Bend your right knee and bring your right foot to the top of your left thigh, as close to your hip as possible.
3. Bend your left knee and bring your left foot to the top of your right thigh, as close to your hip as possible.
4. Place your hands on your knees or in your lap, and lengthen your spine.
5. Breathe deeply and hold the pose for several breaths, or as long as comfortable.




Mediapipe Feed