All the programming has been done in python programing language except the Haar cascade training module which uses C++. The platform used for python programming is Jupyter Notebook using Anaconda3 64-bit.

**Image Segmentation using thresholding**

Segmentation means separating portions (segments) of an image from other segments of the image. It is mainly done in order to extract features from an image by keeping the useful information from the image and discarding the rest of the parts.

In thresholding, every pixel of the image is compared to a threshold value and if the pixel value is smaller than the threshold then the pixel is set to ‘0’ else it is set to one. Thus, pixels of the whole image are divided into two regions- the ones with pixel value greater than the global threshold and those with pixel values less than the threshold. The OpenCV library has various functions for thresholding, the most common one being cv.threshold().

There are multiple ways of performing segmentation in images some of which are mentioned below:

* Supervised Segmentation by thresholding- Manual input

In this method, the user has to manually give a threshold value and all the pixels in the image are segmented based on that single threshold value. Such a threshold is called Global Threshold.

* Adaptive thresholding

Global thresholding might not be good in some cases especially when the lighting is not uniform throughout the image. This may give undesired segmentation results. To avoid this, adaptive thresholding is used. In adaptive thresholding, threshold are locally determined for small regions of the image and thresholding is done accordingly. This gives far better results than global thresholding in images with varying illumination.

* Otsu’s Binarization

In Otsu’s method, threshold is determined automatically unlike in global thresholding. Otsu’s algorithm tries to find a threshold value which minimizes the weighted within-class variance.

In this project, it was tried to perform segmentation using manual thresholding on images in the attempt to separate the background floor area from the human in the images. Once the human is separated, one can perform the further steps of detection more easily and accurately.

OpenCV function with a global threshold value of 50 was used in this project.

**Method:**

**Step 1:** Conversion from BGR to GreyThe coloured BGR (Blue,Green,Red) image is converted so that the thresholding can be done with reference to a single grey channel instead of comparing all the three colour channels separately.



**Step 2:** Thresholding by using a global threshold of 50.

Various values of threshold were tried and the results were examined. The threshold of 50 was found to be best to separate the human from the background.



**Step 3:** Segmentation based on thresholding.

Now, the image is divided into two segments depending on the result of thresholding.

**Results:**

The result of segmentation was not as expected and didn’t fulfil the requirement. The following disadvantages were found:

* The thresholding depends highly on the lighting conditions of the room. Also, different portions of the image did not receive equal light and shadows of objects and human made segmentation difficult.
* The colour of the garments that the human is wearing also matters a lot in thresholding. When the colour of the garment is similar to that of the floor, segmentation becomes very difficult.
* Different coloured floors will need different threshold values. Also, other objects with complex structures and colours can make the segmentation different.

Because of these limitations, segmentation of image to remove background from the image was not found reliable and was not used any further.

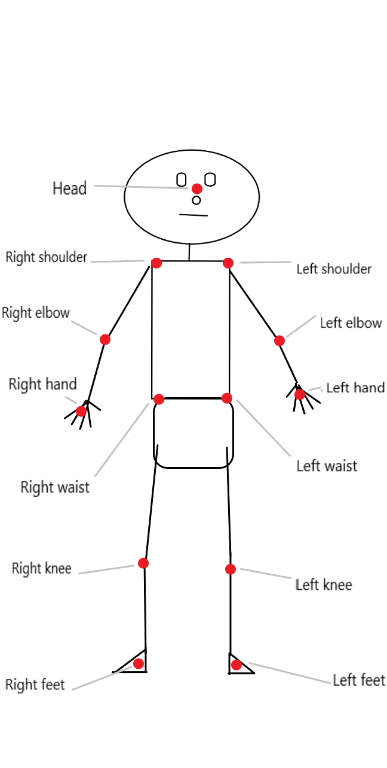
**Body landmark detection**

Human body has a number of landmarks with which one can easily perform pose estimation. These landmark include major joints and features of the body including knee, elbow, hip, etc. This information can be used to detect if a person has fallen.

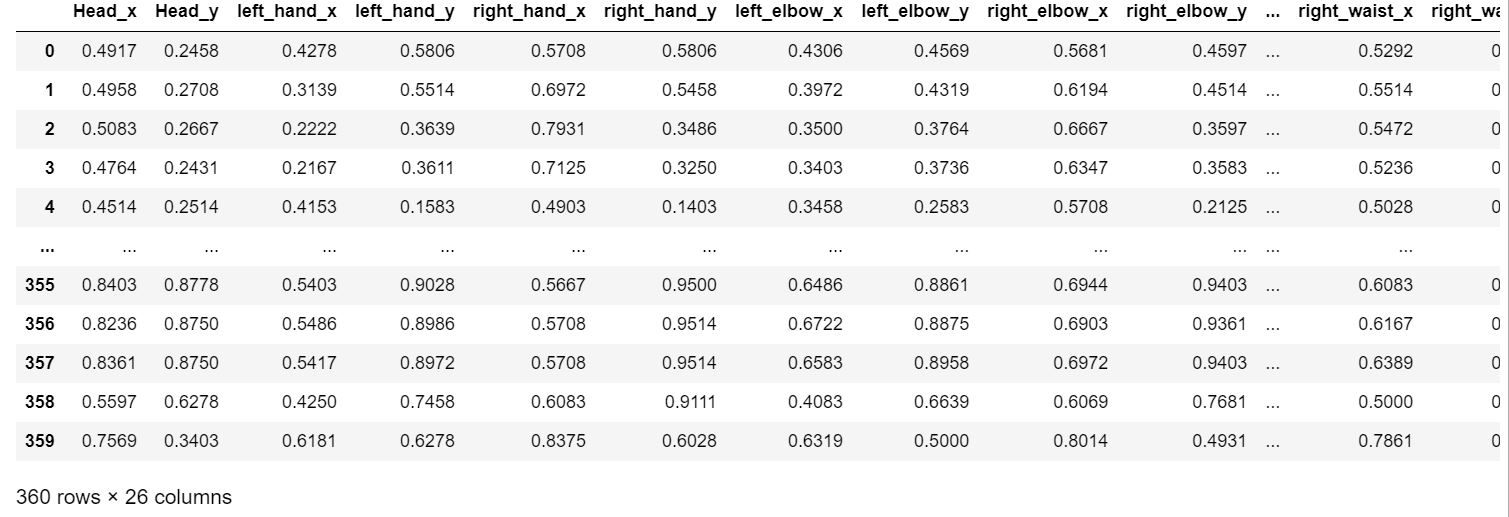
There are many body landmark detection models which have been proposed, one of which is Google’s Mediapipe Pose model (BlazePose GHUM 3D). BlazePose is a state-of-the-art pose estimation model which predicts the location of 33 body landmarks ranging from eyes, nose, hip, to toes. It is highly accurate and achieves real-time performance on most of modern devices including smartphones, desktops/laptops and even web.

In order to study how the body landmarks were detected, a CNN model was trained with the objective to detect 13 body landmarks which used regression to estimate the x and y coordinates of respective landmarks for all the landmarks.

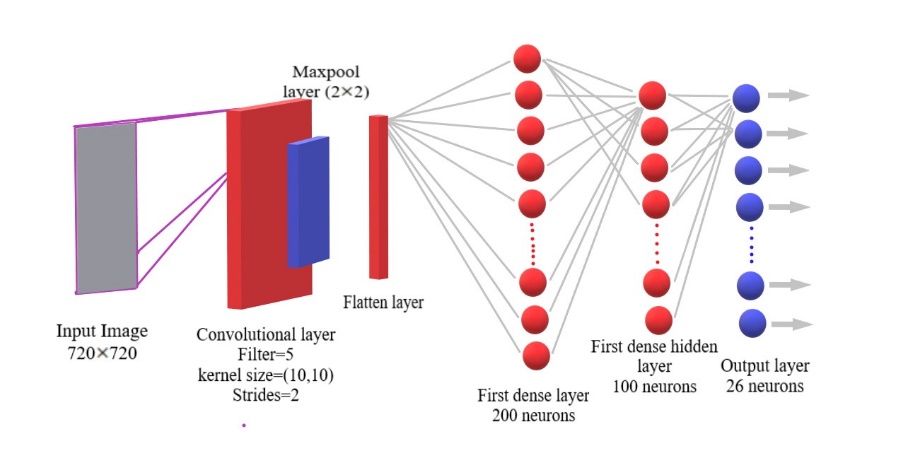
Here are the 13 body landmarks that were used for pose estimation:

1. Head
2. Left hand
3. Right hand
4. Left elbow
5. Right elbow
6. Left shoulder
7. Right shoulder
8. Left waist
9. Right waist
10. Left knee
11. Right knee
12. Left foot
13. Right foot

The CNN was trained using a custom dataset consisting of 360 images with a resolution of 720720 pixels. Every image contains human in different postures depicting various activities of a workout routine. The landmarks for all the images are collected manually one by one which consists of x and y coordinates of every landmark. The landmarks were then normalized by dividing every value by 720 so that all the values lie in between 0 and 1.



The CNN model consisted of a convolutional layer with a depth of 5 followed by a MaxPooling layer of dimension 22. Then the output of the MaxPooling layer is flattened and sent to dense ANN network. The ANN network consists of 3 layers. The first layer has 200 neurons, the second layer had of 100 neurons and the last layer, which is also the output layer, has 26 neurons, one for each coordinate value. Since the model is based on regression, all the dense layers have ReLU (Rectified Linear Activation) function.



For training, the 720720 resolution images were given as input to the CNN layer while the corresponding outputs which are coordinates for the landmarks were also given for supervised learning.

The dataset, which consisted of 360 images and corresponding landmarks, was divided into train and test samples using Sci-kit learn’s test-train split in a test to train ratio of 8:2. The model was trained for 5 epochs and the model was then tested on test images. The results found were as follow:

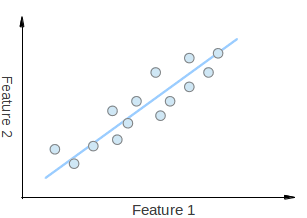


**Conclusion:**

The model was tested on test input images and the accuracy was found to be very low. The estimated landmarks were not in agreement with the actual landmark positions.

The possible reason for the failure of the model can be the incorrect choice of hyper-parameters of the neural network i.e. structure and depth of the convolutional layers, size of the dense layers, the activation function, etc. Also, the task of landmark estimation may be too complex for this rather simple model and it might have led to underfitting. Further experimentation and study will be required to realize an accurate pose estimation model using body landmarks. Google’s BlazePose is highly advanced pose estimation model which is fast as well as accurate.

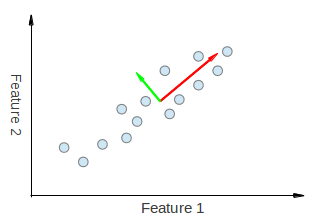
**Principle Component Analysis**

Principle Component Analysis (PCA) is a statistical method that is used to extract the most important features of a sample. PCA is very useful for dimensionality reduction which is the process of reducing the dimension of a dataset. The following is an example of dimensionality reduction:

Here, we have a 2-dimensional dataset with points which vary along the two axes- feature 1 and feature 2. Although the points may seem random but from observation it can be noticed that the points are somewhat along a straight line which is represented by the blue line.

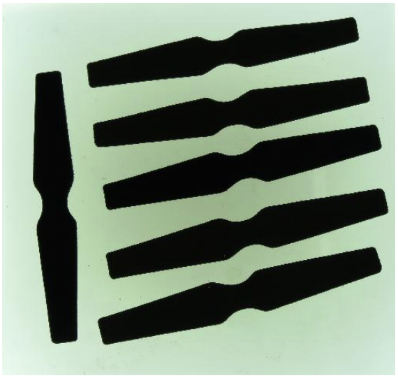
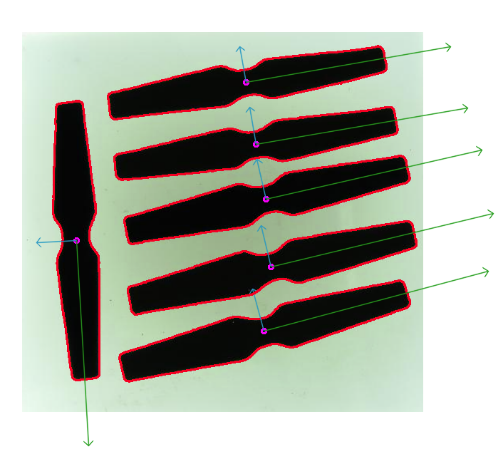
Moreover, knowing where a point lies on the blue line will give more information than knowing where it lies on any of the two axes. So, the information that otherwise needed two dimensions can is now somewhat given in one dimension too. This leads to dimensionality reduction.

PCA gives the direction along which the data varies the most. It does so by giving vectors which are most important (principle) features and are called eigenvectors. These eigenvectors contain multiple information regarding how the data is spread across the dimensions. The length of any eigenvector represents how much the data varies along its direction. Also, all the eigenvectors start from a common point which is the centre of all the points in the dataset.



Principle component analysis can be used to find the orientation of objects in an image. The eigenvectors will give the directions of maximum spread of object and the direction perpendicular to it.

Here is an example on how PCA helps determining the orientation of objects in an image:



In human fall detection, the orientation of the human in an image is a useful feature to determine the posture and detect falls. For this purpose PCA was tested on the database images and the results were observed.

Here are the steps required for finding object orientation from images:

1. First of all the region of interest is chosen by some object detection technique. In this project it was done using YOLO object detection which gives the bounding box enclosing the object which in this case is a human.



One of the challenges is that the edges of the image are also detected as contours which give false contours. To avoid this, the image is added with some padding on the edges and then the edges are blurred.

1. First of all the image is converted from BGR to grey. This is done for performing thresholding on the image with an appropriate threshold value. This gives out a binary image.
2. Next, contours are detected from the binary images. These contours are the boundaries of the objects which are differentiated by sudden gradient changes.
3. These contours are then tested for desired size. Contours enclosing a certain range of area are kept while those smaller or larger than those are discarded.
4. After the desired contours are extracted, PCA is performed over those contours to find the eigenvectors. These eigenvectors are then scaled properly to represent the variance of data and then these eigenvectors are superimposed on the original image for visualization.

**Results:**

There were some issues with PCA which restrain its use for finding orientation of human body in an image-

The determination of contours depend on the gradient changes in the binary image, which in turn depends on the relative colours of the human, surrounding, clothes, and also on the lighting. Due to this, the human as a whole is not detected as one contour. Rather, different parts of the body and clothes are detected and processed as separate contours which don’t serve the purpose of finding the orientation of the human body.

For this reason, orientation of the human was not determined using PCA and another algorithm had to be used which is discussed in a later part of this report.