**Dependencies**

* Python3
* Tensorflow 1.3.6, Tensorflow Graphs
* Opencv3, python3-tk, protobuf
* Build c++ library

**Requirements file**

* [argparse](https://docs.python.org/3/library/argparse.html)
* Matplotlib
* SciPy
* Dill
* TQDM
* Fire
* Requests

**pip install -r requirements.txt** *#For installing the requirements*

After installing all the mentioned requirements,

**ESTIMATE THE 3D POSE IN REALTIME**

1.Click on **Run the application→’tf-pose’ button** in GUI to launch the command window. Once the command window prompts up, navigate to the project directory folder. The file testwebcam3D.py is located under the directory *tf-pose\src\ testwebcam3D.py*. Executing this file will open up real-time webcam. Then execute the below command,

**python testwebcam3D.py** #*Execute this python file for Realtime Webcam*

Now a scenario where a person walks Infront of the camera. For instance, we are conducting an experiment of normal walking. The person may be a Normal/Abnormal person. The person walks for a time limit of 10 sec approximately, where certain number of frames are captured.

2.Post this scenario, extraction of coordinates values takes place. The key points are collected for the corresponding frames and are stored in a text file called pose\_3d. An automation is done here where we write the coordinate value in a file(pose\_3d). Previously the coordinate values have to be copied from the command prompt and have to be post processed in order to get a proper format (This was previously time consuming). Now we have the proper file format and we are ready to calculate three gait parameters.

3.Then click on the button **Extract the gait features → “Load”** to extract the necessary features from the file for further processing. The features are Step length, Step width and Gait speed. For the measures - frequency and step time, we need to get complete human coverage during walking and also frame rate needs to be computed. The package we are using is 3D pose using TensorFlow and OpenCV and it has limitation of covering only up to 3-4 meters. Hence, we proceeded only with mentioned 3 metrics. After figuring out the 3 gait parameters, we are finding the Mean, Standard deviation, Minimum and Maximum values for the measures as well. So, we are converting the test records in to a proper format of what we have in the training data. This entire processing is done via a python code which is explained in detail in the upcoming section.

4. Now the calculated data is consolidated and stored in a data frame for further evaluation. **Then click on the “Submit” button**. This will help to evaluate the test and training records using the evaluation algorithms defined. Logistic regression and naïve bayes are evaluating algorithms used. With the already trained model, we will now evaluate the test records and obtain the accuracy. to get the results displayed on the console window of Spyder.

The console will display the following: Gait parameters: Step length, Step width, Gait speed. The class of the persons: 1- Normal, 0- MCI. Accuracy: The test records will be evaluated with the trained model (Normal, Counting Backwards 3 (CB3), Counting Backwards 7 (CB7) etc.) using the algorithms.

**TEST INTERFERENCE, FROM IMAGES, WE ESTIMATE THE 3D POSE**

1.Click on **Run the application→’tf-pose’** button to launch the command window. The command window will open. Now navigate to the project directory location. *\tf-pose\src\runworking.py .* Leave the select the option drop down button as blank. We are going to upload an image of a person in jpg. Once we execute the run file using the below command, we get all the 2D key points and those are connected with best possible straight lines.

**python runworking.py --image==exampleimg.jpg**

2.We get all the 2D key points and those are connected with best possible straight lines.

3.An automation is done here where we write the coordinate values directly in a file(img\_3d.txt

4.We also get the *heat map*, Vector map-x and Vector map-y along.

5.Finally using these 2D key points we estimate the 3D pose from Images.

**Main Repository used**-- <https://github.com/ildoonet/tf-pose-estimation>

**Repository used for 3D plotting**-- <https://github.com/pyqtgraph/pyqtgraph>