**STEPS TO WORK WITH UBER DATA**

1.At first the UBER data is imported as individual column vectors with the trip-id, latitude, longitude as vectors of type double and the time stamp as a cell array.

2.At first, we have to run the **time\_slot\_creation.m** with timestamp as input and it returns the timeslots(time-data) as output.

3.Then we need to horizontally concatenate the trip-id, time data, latitude, longitude to form a matrix (say **A**) having 4 columns.

4.Then we need to pass the matrix **A** as well as the cell array of time stamp as input to **segmentation.m** and it removes the tripids and timestamps associated with it which are outside the desired bounding box of latitude and longitude.

5.Then we have to pass the matrix **A** to the **averagetravel.m** to get the average change in latitude and longitude traversed by cabs in 20 secs.

6.Then we have to run the script **centre.m** to get the coordinate of the centres.

7.Then we have to run the function **centre\_finder\_1.m** to get the closest centre associated with each coordinate.

8.Then we have to run the **road\_namer.m** to create record of each road segment between the centres.

(**NOTE:** AFTER THIS WE HAVE TO REMOVE THE COLUMNS OF LATITUDE AND LONGITUDE FROM MATRIX **A** SO THAT MATRIX A IS IN FOLLOWING FORMAT

TRIPID/TIMESLOTS/CENTRES)

9.The temporal data is obtained from the function **temporal\_finder.m**

10.Then we have to run **centre\_finder\_manipulate.m** to visualize each trip data more simply and also to find the travel time.

11.Then we have to run **uber\_matrix\_creator.m** to search the road segment for each trip from road list and assign it.

12.Then we have to pass the output of previous function to **timeslot\_adder.m** to add timeslots to the corresponding trip-id.

13.Then we have to pass the output of previous function to **road\_modifier\_1.m** and then output of previous function to **road\_modifier\_2.m** to get the modified matrix.

14.Then to get geographical features we need to run three functions **neighbour\_checker.m, roadlength.m, bidirectional\_checker.m** and then concatenate output of each three functions to get the geographical matrix.

15.Next we need to download tensor tool box and form the sparse tensor using **sptensor** command.

16.After this we need to decompose the tensor using Tucker Decomposition by running the function **tucker\_als** from the Tensor Tool Box and then reconstructing the tensor using tensor toolbox.

17.Finally to find travel time of any path, we need to run the function **concatenate.m** and the output would give the travel time.