**Point Mass Model Documentation**

**Open ‘point\_mass\_main’ to run the LTS codes.**

Currently, coefficient of friction is determined based on past data, and is fixed all time.

In the current codes, I fist assumed maximum speed of car is 120 kmh. Then consider straight line, where Resultant Force = Friction Force – Drag, this formula can be solved when Resultant Force = 0. If put in V = 120kmh, will get coefficient of friction is around 0.2.

**Function: bsp**

This function output the Boundary Speed Profile of the car.

**Function: acceleration**

This function output the available longitudinal acceleration of the car, after providing portion of the friction force to cornering motion.

**Function: accel\_lsp**

This function requires the ‘acceleration’ function.

It alters the data points in bsp after considering car’s acceleration, then output the Limit Speed Profile with the consideration of acceleration.

**Function: local\_finder**

This function helps find the local minima of a 2d array, which is used here to find the local minima of Limit Speed Profile, and hence to determine the braking points.

**Function: brake**

This function help determine, if I know the speed of the next point (point 2), what can be the maximum speed of this point (point 1) be, considering the corner force at point 1.

In the codes, the program test one by one the speed at point 1 and calculate the speed at point 2, until the speed at point 2 hits above the required value.

**Function: lsp**

This function requires the ‘local\_finder’ function, ‘brake’ function.

This function will run after obtaining the Limit Speed Profile with acceleration. It locates the braking points first, From every local minima, it calculates the maximum speed that the previous point can have, and replace it.

The loop will stop (LSP will stop being replaced) when the maximum speed is higher than the LSP speed, means that braking is not required anymore.

**Function: Lap\_Time\_Simulation**

Base on the obtained Limit Speed Profile, velocity at every gap is the average speed of previous point and next point. Distance is the gap between that two points. Time intervals are calculated and accumulated to get the total lap time.

**Update:**

The lap time simulation computed by the above functions have around 50% inaccuracy, which I believe is unacceptable. These functions have been reviewed, and the main source of inaccuracy has been identified as the constancy of friction coefficients. The coefficients of friction is unable to be accurately defined, while the same time longitudinal and lateral coefficients can be so different that need to be viewed separately.

Therefore, implementation of tyre model is very important. The following functions are based on alternations of the above, with partially the tyre model built by Ji Hao.

**Function: slip angle**

Slip angle is an important factor to the tyre performance. The slip angle in this model is under the assumption that, the driver at point 1 aims at (steers to) point 3, but the car will actually travel to point 2. Therefore, angle between tangential line at point 1 and line13 would be the steering angle at point 1. Angle between line12 and line13 would be the slip angle at point 1. Angle can be solved using cosine rule.

This function inputs a track model. This outputs a **1d array (a list) of slip angle**.

**Function: tyres**

This function is completed by Ji Hao, and reused in this model again.

It takes in camber, slip angle, normal load, and the tyre model of specific tyre pressure. It outputs the lateral force and longitudinal forces at the given conditions.

**Function: v\_calculater**

This function is to find the maximum speed the car can have under specific conditions.

To find the maximum cornering speed of a car, we should put Lateral Force = Centripetal Force, shifting radius and mass to the LHS and square root will have speed.

In this case, Lateral Force is provided by Function: tyres. It is a function of normal load (weight + downforce), which is also dependent on car speed. Now, we have speed at both sides of equation, it becomes impossible to directly solve it out.

Therefore, iteration is used in this function. From 0 to maximum possible speed, the function calculates Centripetal Force and Lateral Force one by one. Iteration will stop once the Lateral Force is smaller than the Centripetal Force, and the output speed would be the previous iterated speed.

This method limits the accuracy of simulation, and the program runs very slow with this algorithm.

**Function: bsp**

The main idea of Boundary Speed Profile remained unchanged. Slip angle is calculated first given the specific track.

On the track, if the curvature is larger than 300m, we treat it as a straight line and the boundary speed would be the maximum speed of car. Otherwise, the speed will be evaluated using **function: v\_calculater.**