Q2c_VehicleDynamics

April 16, 2024

[ ]:
```
function VehicleDynamics(states, control)
    la = 1.56
    lb = 1.64
    m = 2020
    g = 9.81
    Izz = 4095
    h = 0.6
    mu = 0.8
    x = states[1]
    y = states[2]
    v = states[3]
    r = states[4]
      = states[5]
    ux = states[6]
     f = states[7]
    ax = control[1]
    d f = control[2]

    Fzf =  m *g * (lb/(la + lb)) - (m*h)/(la+lb) * ax #TODO Front axle load
    Fzr =  m *g * (la/(la + lb)) + (m*h)/(la+lb) * ax #TODO Rear axle load

     f =  f - atan((v + la * r)/ux) #TODO Front slip angle
     r = -atan((v-lb*r)/ux) #TODO Rear slip angle

    Fyf = MagicFormula( f, Fzf, mu) #TODO Front lateral force
    Fyr = MagicFormula( r, Fzr, mu) #TODO Rear lateral force

    dx           = ux * cos( ) - v * sin( )#TODO
    dy           = ux * sin( ) + v * cos( )#TODO
    dv           = ((Fyf + Fyr)/m) - ux * r#TODO
    dr           = (Fyf * la - Fyr * lb)/Izz#TODO
    d           =  r #TODO
    dux          = ax #TODO
    d           = d f #TODO
    dstates      = [dx dy dv dr d  dux d ]
    return dstates
end
```

```julia
function MagicFormula(alpha, Fz, mu)
    B =     5.68 #TODO Input Q2b value here
    C =    1.817 #TODO Input Q2b value here
    Fy =  mu * Fz * sin(C * atan((B/mu) * alpha))  #TODO Lateral force
 ↪calculation
    return Fy
end


x0 = [-10.0 -5.0 0.5 0.1 0.1 10.0 0.1] # This is the initial state
ctrl = [1 0.1]  # One step control action
dstates = round.(VehicleDynamics(x0, ctrl); digits = 3) # Calculate states
 ↪derivative
println("The states derivative is: ", dstates)
```

```
The states derivative is: [9.9 1.496 -1.004 2.587 0.1 1.0 0.1]
```