Torque vectored stability control for in-hub electric motorized vehicles

Stephen Chen

May 5, 2012

1 Introduction

With lightweight, coreless flux motors, vehicles can possess individually driven wheels using a hub motor mount topology, or some similar individually driven configuration. Such design is not completely unheard of, as examples in Mercedes's SLS AMG E-CELL and Audi's R8 E-TRON can be found.

With this ability to define individual drive characteristics to each wheel, dubbed "torque vectoring," vehicles exhibit excellent handling characteristics, as it can generate yaw moments by regulating the torque at each wheel. The yaw moment can be controlled then not only by steering and differential braking, but additionally through torque vectoring.

This project attempts to utilize the handling characteristics of hub motor mounted electric vehicles to implement a torque vectored stability (yaw moment) control.

2 Modeling

2.1 Vehicle modeling

We utilize a 7 degree of freedom model of the vehicle dynamics with the following equations of motion. Note that we do not model roll or weight transfer, thus, the normal force F_z at each wheel will not vary with time.

$$m\ddot{x} = (F_{xfl} + F_{xfr})\cos(\delta) + F_{xrl} + F_{xrr} - (F_{yfl} + F_{yfr})\sin(\delta) + m\dot{\psi}\dot{y}$$

$$m\ddot{y} = (F_{yfl} + F_{yfr})\cos(\delta) + F_{yrl} + F_{yrr} + (F_{xfl} + F_{xfr})\sin(\delta) - m\dot{\psi}\dot{x}$$

$$I_z\ddot{\psi} = l_f(F_{xfl} + F_{xfr})\sin(\delta) + l_f(F_{yfl} + F_{yfr})\cos(\delta) - l_r(F_{yrl} + F_{yrr})$$

$$+\frac{l_w}{2}(F_{xfl} - F_{xfr})\cos(\delta) + \frac{l_w}{2}(F_{xrr} - F_{xrl}) + \frac{l_w}{2}(F_{yfl} - F_{yfr})\sin(\delta)$$

$$J_w\dot{\omega}_{fl} = \tau_{m,fl} - r_{eff}F_{xfl}$$

$$J_w\dot{\omega}_{fr} = \tau_{m,fr} - r_{eff}F_{xfr}$$

$$J_w\dot{\omega}_{rl} = \tau_{m,rl} - r_{eff}F_{xrl}$$

$$J_w\dot{\omega}_{rr} = \tau_{m,rr} - r_{eff}F_{xrr}$$

To determine the proper lateral and longitudinal tire forces, we need to determine the lateral slip angle and longitudinal slip ratio, as the force from a tire is a function of tire slip. We define α to be the tire slip angle, and σ to be the tire slip ratio. Subscripts of f and r will therefore refer to front or rear, respectively.

The tire longitudinal slip ratio in fact comprises a piecewise function, chosen based upon acceleration or braking, to avoid a singularity. We choose to simply use one, since we assume our car is in and stays in motion. The longitudinal slip for an individual tire is defined then as

$$\sigma = \frac{r_{eff}\omega - \dot{x}}{\dot{x}}$$

Here ω is a placeholder variable for any of the individual wheel speeds.

The tire lateral slip angle is given based upon front or rear location, as the steering angle δ will affect it. The slip angles are given by

$$\alpha_f = \delta - \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}}$$

$$\alpha_r = -\frac{\dot{y} - l_r\dot{\psi}}{\dot{x}}$$

With these definitions for lateral and longitudinal slip angles and ratios, we then require a tire model utilizing these parameters and effectively mapping the slip to the force. For this project, we choose to utilize the "Magic Tire" formula, which is represented as

$$F(z) = D\sin\left(C\arctan\left(B(1-E)z + E\arctan(Bz)\right)\right)$$

z refers to the slip input (ratio or angle in degrees), and the constants B, C, D and E are chosen by fitting curves.

For our vehicle model, we assume the center of gravity to be exactly at the middle of the car, which is not completely unrealistic in this specific case, as the battery and motor weights can be easily dispersed evenly through the car.

2.2 Motor modeling

The motor is modeled in a simplified fashion which can be generalized across different motor topologies: DC brushed, DC brushless, 3-phase permanent magnet (axial and radial flux), etc. The motor is viewed to have a series resistance and a back-emf generator.

The motor is assumed to have a torque constant and back-emf generation constant, that is, the torque from the motor τ_m is proportional to the current through the motor armature i, and the back-emf e is proportional to the motor angular speed ω. This relation is realized in the set of equations

$$\tau_m = k_t i \quad , \quad e = k_e\omega$$

where k_t is the torque constant, and k_e is the back-emf constant.

The electric dynamics equations follow from the circuit, and are formulated as

$$V - ir_m - e = 0$$

where V is our control, the voltage across the motor, and r_m is the resistance in the motor armature. From our assumed relations above, we substitute to find

$$V - \frac{\tau_m}{k_t}r_m - k_e\omega = 0$$

Rearranging,

$$\tau_m = k_t\frac{V - k_e\omega}{r_m}$$

In practical implementation, varying the voltage V is done through PWM following some sort of converter/inverter. However, for the intent of this project, it is assumed that the voltage applied can be arbitrarily chosen.

3 Control design

3.1 Upper level controller for vehicle stability

We use a driver model where the driver's input (the steering angle) is used as our reference for desired yaw rate. We assume the vehicle has Ackermann steering geometry, as well has neutral steer. Thus, the curvature of the turn, modeled as a circle with radius R, is given as

$$\frac{1}{R} = \frac{\delta}{L}$$

where δ is the steering angle and L is the wheelbase of the car. The desired yaw rate will then be given as

$$\dot{\psi}_{des} = \frac{\dot{x}}{R} = \frac{\dot{x}}{L}\delta$$

Since we are dealing with a highly nonlinear system, we choose to use sliding mode control as opposed to linear control methods. We choose our sliding surface s to be

$$s = \dot{\psi} - \dot{\psi}_{des}$$

Utilizing this, we find our control law of the form

$$M_\psi = \frac{I_z}{1+\cos(\delta)}\left(-\frac{l_f}{I_z}(F_{yfl}+F_{yfr})\cos(\delta) + \frac{l_r}{I_z}(F_{yrl}+F_{yrr}) - \eta s\right)$$

where M_ψ is the yaw moment torque we desire to generate.

3.2 Lower level controller for motor torque generation

From our wheel dynamics, we can infer the necessary torque we need to drive at each wheel. Since we can define M_ψ in terms of longitudinal wheel forces, we can determine the torques we need to apply at each wheel. We assume that the driver has some input torque (determined by the throttle pedal position), and we modulate about that torque. We choose the torques to be differential from left to right, and are

$$\tau_l = \tau_0 + \frac{r_{eff}}{l_w}M_\psi$$

$$\tau_r = \tau_0 - \frac{r_{eff}}{l_w}M_\psi$$

where τ_0 is the driver torque input. We drive the left side motors to τ_l and conversely drive the right side motors to τ_r to generate our desired yaw moment torque.

The actual torque is estimated from the current through the motor armature, which is typically measured using a series sense resistance or magnetic hall effect sensor. This current is then scaled by the torque constant to obtain our estimated torque.

We chose a proportional-integral (or PI) controller to reach our torque setpoint. The control law has the form

$$u(t) = k_p(\tau_{m,des} - \tau_{m,est}) + k_i\int_0^t(\tau_{m,des} - \tau_{m,est})dt$$

where $u(t)$ is our control signal (i.e. the applied voltage V), $\tau_{m,des}$ is our desired torque, and $\tau_{m,est}$ is our estimated (sensed) torque.

Since the electric dynamics converge much faster than that of the physical constants, we assume for the simulation that $\tau_m = \tau_{m,des}$. In doing this, we also generalize our control algorithm for not only electric motor drive, but for any torque vectoring drive system.

4 Simulation results

4.1 Torque vectoring verification

To verify that our all wheel drive in wheel electric motor drive system can in fact affect the yaw moment, we utilized a simulation where we can independently define torques on any given wheel. By choosing torques, we can create certain trajectories without using a steering input (setting $\delta = 0$). The simulation provided the following results for a yaw moment generated purely by differential torques.

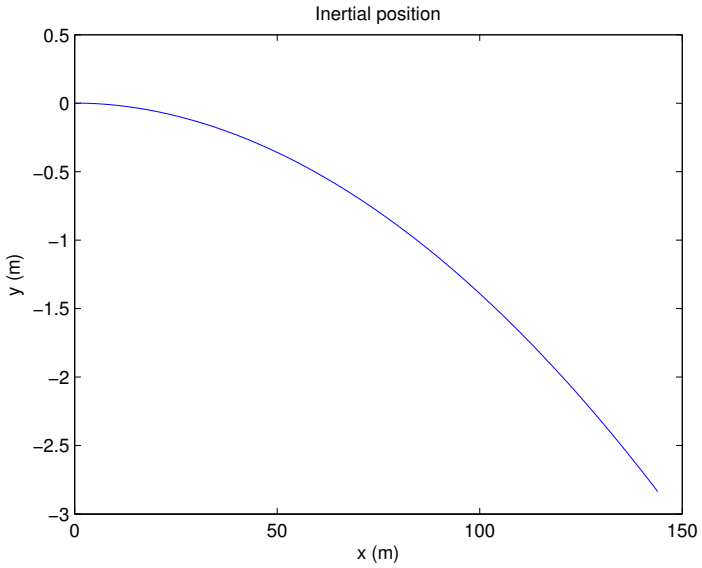


Figure 1: Vehicle trajectory generated with $\delta = 0$ and torque vectoring

With an inertial coordinate initialized at $(x, y) = (0, 0)$, we can see that utilizing a saturated torque input on two wheels and a zero torque input on the other two can cause a significant change in yaw moment.

4.2 Control verification

We use a driver model that has a constant steering angle δ and constant torque input τ_0. For our initial controller with sliding parameter $\eta = 20$ and $\mu = 0.9$,

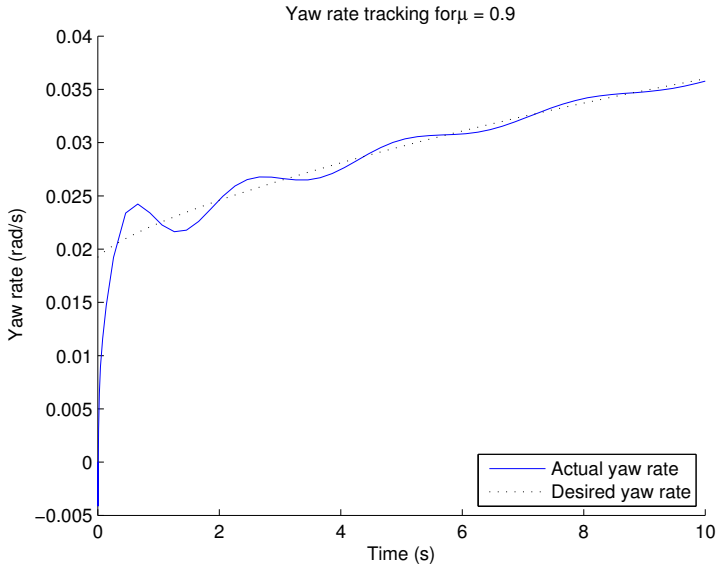


Figure 2: $\dot{\psi}_{des}$ tracking for dry conditions, $\mu = 0.9$

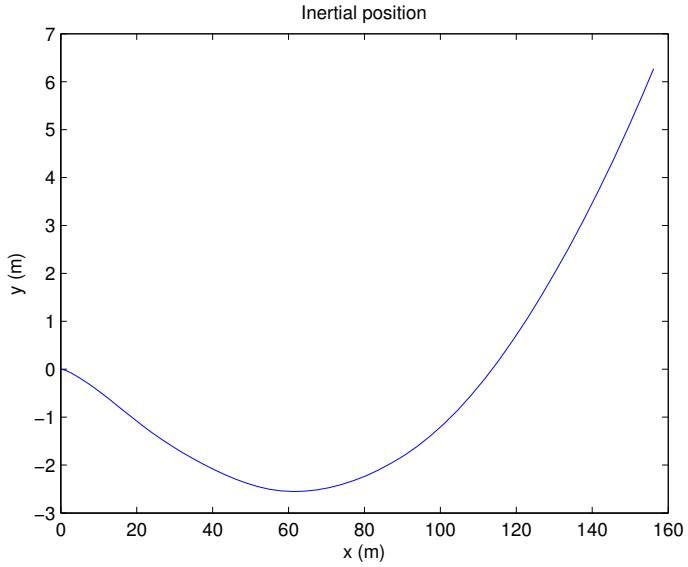


Figure 3: Inertial position for controlled $\mu = 9$ conditions

We notice an oscillatory response around our desired tracking. However, this is minuscule and should not matter much. In dry conditions we can see the response is tidy and tracks rather well.

For the same controller but on wet conditions ($\mu = 0.5$), we can see the stability controller come into play.

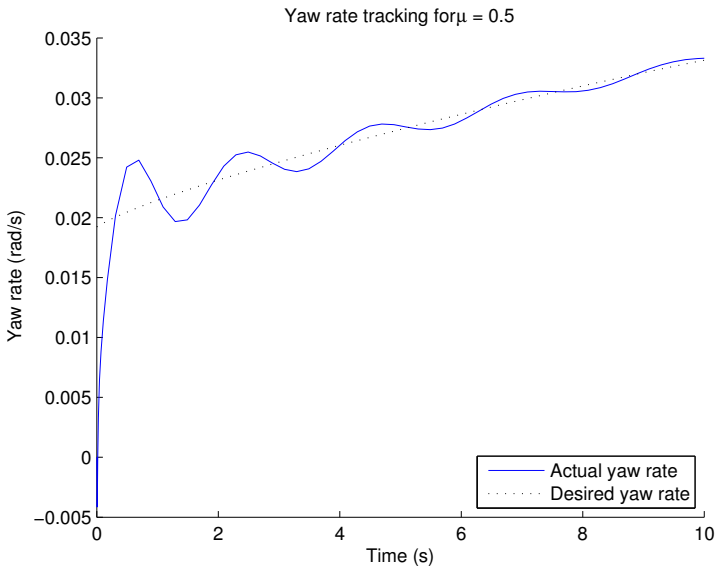


Figure 4: $\dot{\psi}_{des}$ tracking for wet conditions, $\mu = 0.5$

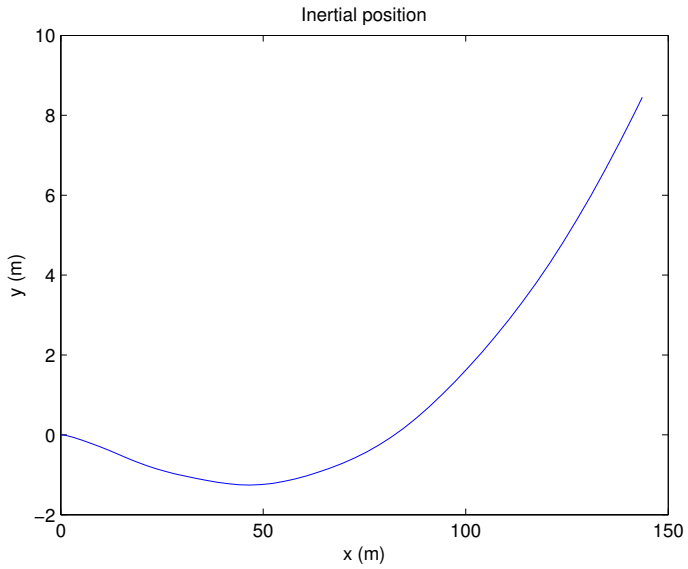


Figure 5: Inertial position for controlled $\mu = 5$ conditions

Here, we can see the oscillations have larger magnitude than those of the dry road, which is expected as the tires cannot provide as much cornering forces to drive the yaw rate back as fast.

Finally, let us consider icy conditions, where roads have been iced over ($\mu = 0.2$). Here, we can note that the stability controller works as a large improvement.

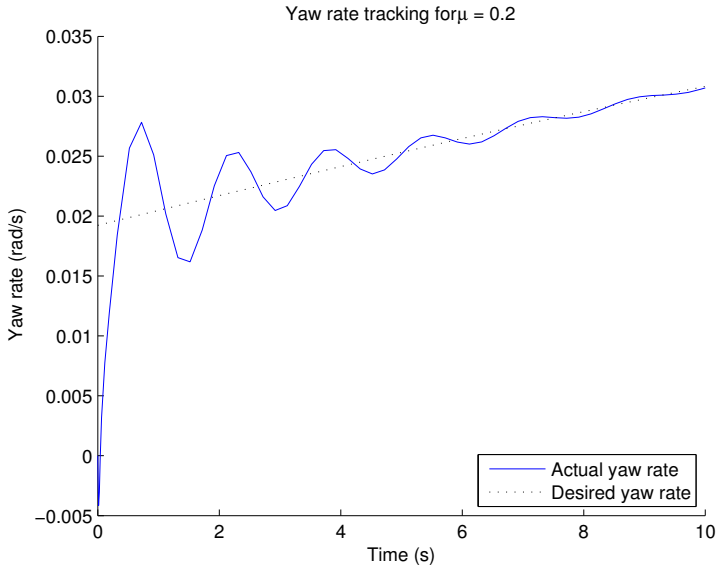


Figure 6: $\dot{\psi}_{des}$ tracking for icy conditions, $\mu = 0.2$

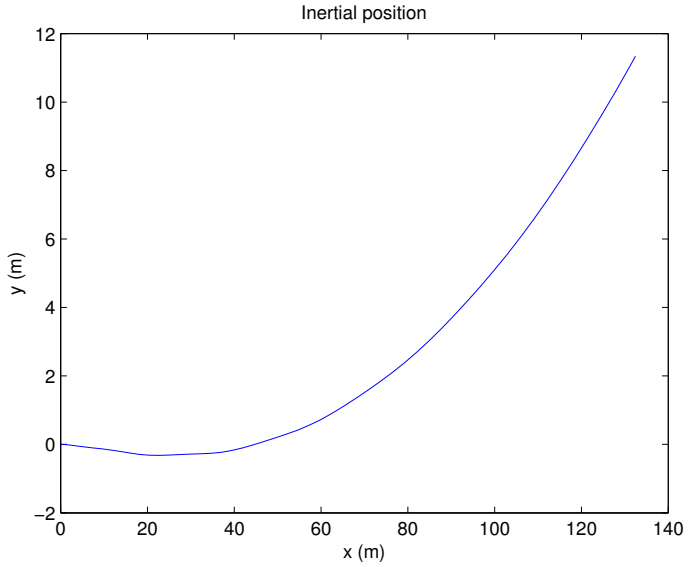


Figure 7: Inertial position for controlled $\mu = 2$ conditions

The stability controller does a relatively excellent job tracking the desired yaw rate on ice. The oscillations again grow, but it is expected even more such that on icy conditions close to no grip will be available. The torque vectoring allows for the desired yaw rate to be approached even while acceleration, even on ice. Of course, there is some physical maximal friction limit for the tires, but the torque vectoring allows for a beneficial utilization.

5 Conclusions

By utilizing torque vectoring, stability can be achieved (to a certain degree) in acceleration and cornering. In conjunction with differential braking, torque vectoring allows for stability in both acceleration and braking phases.

While the primary intent of this project was determining the mobility and stability control of differential in wheel drive vehicles, the stability control has been simplified such that it is applicable to any form of torque vectoring. Likewise, while a controller was designed for torque control of an electric motor, it was not utilized in simulation due to additional overhead and complexity. Further research should be pursued to see this additional complexity and its effect on the stability of the system, although it is speculated to not have a very large affect. Similarly, other paths to pursue include using a more descriptive model, as the vehicle model utilized does not take roll or pitch dynamics into account.

6 Appendix A: MATLAB m-code

```
% Stephen Chen
% ME131 project m-file
% Stability control for torque vectored in-hub electric motor vehicles

close all;clear all;clc;
```

Parameters

vehicle parameters

```
car.m = 2000; % car mass, kg
car.g = 9.81; % gravity acceleration
car.reff = 0.3; % tire effective radius, m
car.jw = 3; % rotor and tire moment of inertia, kg m^2
car.iz = 5000; % car moment of inertia about cg, kg m^2
car.lf = 1.3; % distance from front tires to cg, m
car.lr = 1.3; % distance from rear tires to cg, m
car.lw = 1.5; % track width, m
car.mu = 0.9; % tire-road friction coefficient
car.fz = 0.25*car.m*car.g; % normal force for one tire
car.nu = 20; % sliding control parameter

% single motor parameters
motor.rm = 0.5; % motor armature resistance, ohms
motor.kt = 0.9; % motor torque constant, N-m / amp
motor.ke = 0.005; % motor back-emf constant, V / (rad/s)

% initial conditions
x0.xd = 0; % initial longitudinal velocity
x0.yd = 0; % initial lateral velocity
x0.psid = 0; % initial yaw rate
x0.omega = [x0.xd / car.reff;x0.xd / car.reff;
x0.xd / car.reff;
x0.xd / car.reff]; % initial wheel speed

% simulation parameters
```

```
t = (0:0.01:10)';
delta = [t 0.005.*ones(length(t),1)]; % steering angle, radians (scalar)
tau = [t 20.*ones(length(t),1) 100.*ones(length(t),1)
 20.*ones(length(t),1)
  100.*ones(length(t),1)];
   % motor torques, N-m
   (4x1 vector: [fl,fr,rl,rr]')
```

Simulation for verifying torque vectoring affecting yaw moment

```
sim('dynamics_2.mdl');
```

Data analysis for verificiation of torque vectoring

Inertial position, x-y

```
figure;
plot(x_i.signals.values,y_i.signals.values);
xlabel('x (m)');
ylabel('y (m)');
title('Inertial position');
```

Simulation for control

```
tau_ref = [t 100.*ones(length(t),1)];
sim('dynamics.mdl');
```

Controller analysis

Psi_des tracking

```
figure;
hold on
plot(psi.time,psi.signals.values);
plot(psi_des.time,psi_des.signals.values,'k:');
xlabel('Time (s)');
ylabel('Yaw rate (rad/s)');
title('Yaw rate tracking for \mu = 0.9');
legend('Actual yaw rate','Desired yaw rate','Location','southeast');
hold off

figure;
plot(x_i.signals.values,y_i.signals.values);
xlabel('x (m)');
ylabel('y (m)');
title('Inertial position');
```

Changing friction

```
car.mu = 0.5;
sim('dynamics.mdl');
```

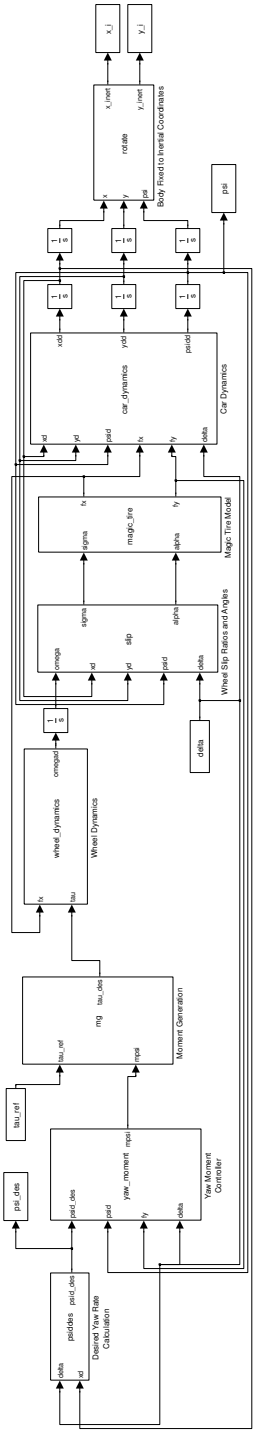
```
figure;
hold on
plot(psi.time,psi.signals.values);
plot(psi_des.time,psi_des.signals.values,'k:');
xlabel('Time (s)');
ylabel('Yaw rate (rad/s)');
title('Yaw rate tracking for \mu = 0.5');
legend('Actual yaw rate','Desired yaw rate','Location','southeast');
hold off

figure;
plot(x_i.signals.values,y_i.signals.values);
xlabel('x (m)');
ylabel('y (m)');
title('Inertial position');

car.mu = 0.2;
sim('dynamics.mdl');

figure;
hold on
plot(psi.time,psi.signals.values);
plot(psi_des.time,psi_des.signals.values,'k:');
xlabel('Time (s)');
ylabel('Yaw rate (rad/s)');
title('Yaw rate tracking for \mu = 0.2');
legend('Actual yaw rate','Desired yaw rate','Location','southeast');
hold off

figure;
plot(x_i.signals.values,y_i.signals.values);
xlabel('x (m)');
ylabel('y (m)');
title('Inertial position');
```


x_I
y_I
x_inert
y_inert
rotate
Body Fixed to Inertial Coordinates
psi
1/s
xdd
ydd
psidd
car_dynamics
Car Dynamics
sigma
alpha
magic_fire
Magic Tire Model
omega
vd
yd
psid
slip
Wheel Slip Ratios and Angles
delta
omegad
wheel_dynamics
Wheel Dynamics
tau_ref
mg
tau_des
Moment Generation
mpsi
psi_des
psid_des
yaw_moment
Yaw Moment Controller
psiddes
Desired Yaw Rate Calculation