07-December-2020

Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

Four-Wheel Steering (4WS) of an Automobile Using LQR and MPC

Abstract:

This document provides a detailed description of the project that was undertaken for the "*EGR 560 Vehicle Dynamics and Controls*" course. The project is based on the lateral control of an automobile by controlling the steering angle of its front and rear wheels. This concept is generally referred to as Four Wheel Steering (4WS) (Not to be confused with Four Wheel Drive – 4WD!).

Basic project description is provided in Section I, followed by the derivation of the dynamics equation and state space form in Section II. Section III performs some system analysis on the open loop plant, while Section IV and Section V provide background information on LQR Servo and MPC controllers. Section VI describes CarSim and the high-fidelity dynamics co-simulation that it provides with MATLAB. Simulation setup and results have been provided in Section VII, and closing comments and future work is present in Section VIII.

Section I: Project Description

A Four-Wheel Steering (4WS) vehicle is an automobile whose front and rear wheel steering angles can be controlled. Some long chassis high-end luxury or sports cars use this to provide better handling at high speeds on highways and improved turning radius at lower speeds within city limits.

By pointing the front and rear wheels in the same direction at high speeds, slip ratio is reduced which leads to a feeling of better control of the car. At low speeds, the rear wheels are turned in the opposite direction as the front wheels to reduce tire friction and aid in turning the chassis within a shorter radius. In this project I explore the latter case of controlling the front and rear wheels in opposing directions at low velocity conditions.

The control systems designed for this project consists of:

- Two control inputs: Front steering angle and rear steering angle
- Two outputs: Lateral position and yaw angle
- Reference signals: Lateral position and yaw angle

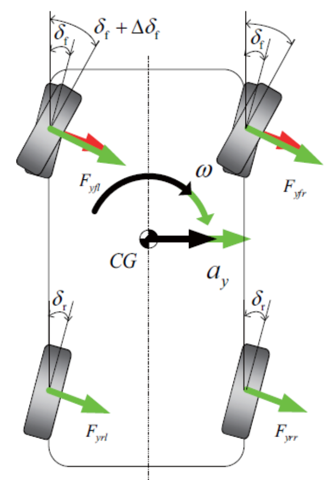


Figure 1: Four Wheel Active Steering Vehicle

The Linear Quadratic Regulator (LQR) setup as a Servo Controller and the Model Predictive Controller (MPC) have been setup to track a set of waypoints consisting of desired lateral position and desired yaw angle.

Section II: System Dynamics

The vehicle being considered is a four wheeled automobile with the steering axles that are able to control the orientation of the front and rear wheels. Some of the steps in the derivation of the dynamics for such a system is provided below:

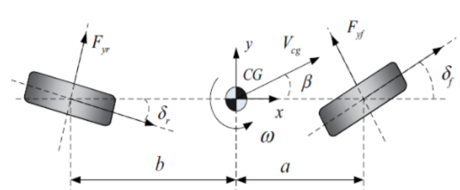


Figure 2: Bicycle Model

Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

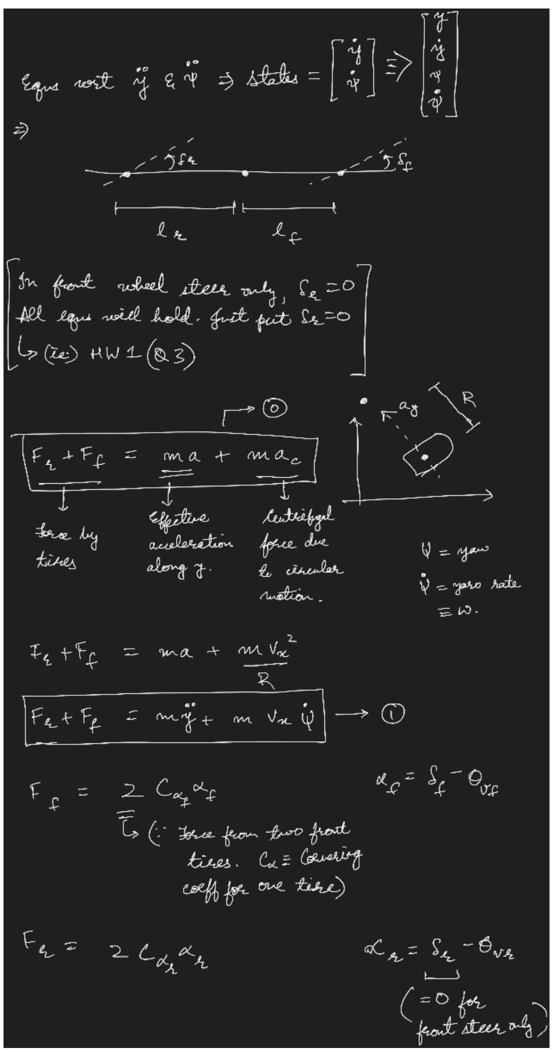
First the four-wheel model (**Figure 1**) is simplified into a bicycle model (**Figure 2**). In reality, the wheel at the inner radius of a turn usually has a slightly larger steering angle than the outer wheel since the arcs traced by them are of varying perimeters. However, for model development to estimate the position of the center of the car, this mild variation in inner and outer steering angles can be considered to be equal. The bicycle model makes an approximation that wheels on either side of the car (left or right) possess the same steering angle while taking a turn.

For all the dynamics equations that follow, the variables, their value and units are provided in **(Table 1)**.

- First, we consider the bicycle model and write the lateral force equations and the rotational torque equation.
- Assuming a linear tire model is used to characterize the tire-road frictional force, we have:

$$\begin{cases} mu\left(\dot{\beta}+\omega\right) = F_{yf}+F_{yr} \\ I_z\dot{\omega} = aF_{yf}-bF_{yr} \end{cases} \qquad \begin{aligned} F_{yf} &= C_f\alpha_f \\ F_{yr} &= C_r\alpha_r \end{aligned} \qquad \begin{aligned} \alpha_f &= \delta_f-\beta-\frac{a\omega}{u} \\ \alpha_r &= \delta_r-\beta+\frac{b\omega}{u} \end{aligned}$$

- Then, the rest of the derivation is done by hand as provided in the image below:



$Y = lateral\ position, \psi = Yaw\ Angle$

Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

$$\Rightarrow \tan(\theta_{vf}) = \frac{\dot{y} + l_f\dot{\psi}}{v_x} \approx \theta_{vf} \text{ for } -20^\circ \le \theta_{vf} \le 20^\circ$$

$$\Rightarrow \tan(\theta_{vr}) = \frac{\dot{y} - l_r\dot{\psi}}{v_x} \approx \theta_{vr} \text{ for } -20^\circ \le \theta_{vr} \le 20^\circ$$

$$\alpha_f = \delta_f - \left(\frac{\dot{y} + l_f\dot{\psi}}{v_x}\right)$$

$$\alpha_r = \delta_r - \left(\frac{\dot{y} - l_r\dot{\psi}}{v_x}\right)$$

From (1);

$$\ddot{y} = \frac{1}{m}\left(F_f + F_r - m v_x \dot{\psi}\right)$$

$$= \frac{1}{m}\left[\dot{y}\left(-\frac{2C_{\alpha f}}{v_x} - \frac{2C_{\alpha r}}{v_x}\right) + 2C_{\alpha f}\delta_f + 2C_{\alpha r}\delta_r + \dot{\psi}\left(\frac{2C_{\alpha r}l_r - 2C_{\alpha f}l_f}{v_x}\right) - m v_x \dot{\psi}\right]$$

$$= \dot{y}\overset{T_1}{\left(-\frac{2C_{\alpha f}}{m v_x} - \frac{2C_{\alpha r}}{m v_x}\right)} + \overset{T_2}{\frac{2C_{\alpha f}}{m}}\delta_f + \overset{T_3}{\frac{2C_{\alpha r}}{m}}\delta_r + \dot{\psi}\overset{T_4}{\left(\frac{2C_{\alpha r}l_r - 2C_{\alpha f}l_f}{m v_x} - v_x\right)}$$

$$I_z\ddot{\psi} = F_f l_f - F_r l_r$$

$$\ddot{\psi} = \frac{1}{I_z}\left[F_f l_f - F_r l_r\right] \quad \left[\text{use same } F_f, F_r \text{ eqns as above}\right]$$

$$= \frac{1}{I_z}\left[2C_{\alpha f}\left\{\delta_f - \left(\frac{\dot{y} + l_f\dot{\psi}}{v_x}\right)\right\}l_f - 2C_{\alpha r}\left\{\delta_r - \left(\frac{\dot{y} - l_r\dot{\psi}}{v_x}\right)\right\}l_r\right]$$

$$= \dot{y}\overset{T_5}{\left(\frac{2C_{\alpha r}l_r - 2C_{\alpha f}l_f}{I_z v_x}\right)} + \overset{T_6}{\frac{2C_{\alpha f}l_f}{I_z}}\delta_f - \overset{T_7}{\frac{2C_{\alpha r}l_r}{I_z}}\delta_r + \dot{\psi}\overset{T_8}{\left(\frac{-2C_{\alpha f}l_f^2 - 2C_{\alpha r}l_r^2}{I_z v_x}\right)}$$

$$\begin{bmatrix}\dot{y}\\ \ddot{y}\\ \dot{\psi}\\ \ddot{\psi}\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0\\ 0 & \boxed{T_1} & 0 & \boxed{T_4}\\ 0 & 0 & 0 & 1\\ 0 & \boxed{T_5} & 0 & \boxed{T_8}\end{bmatrix}\begin{bmatrix}y\\ \dot{y}\\ \psi\\ \dot{\psi}\end{bmatrix} + \begin{bmatrix}0 & 0\\ \boxed{T_2} & \boxed{T_3}\\ 0 & 0\\ \boxed{T_6} & \boxed{T_7}\end{bmatrix}\begin{bmatrix}\delta_f\\ \delta_r\end{bmatrix}$$

A system with states $[y, \dot{y}, \psi, \dot{\psi}]$is thus obtained.

Note: The state equations for this system depend on the longitudinal velocity Vx. If this velocity changes, the system's A matrix will change, thus making time varying. For the purpose of this project, Vx is kept constant to ensure an LTI system.

Table 1: Nominal Parameters (using Class B Hatchback)

Symbol	Name	Value	Unit
m	Mass	1110	Kg
I_z	Inertia (Z axis)	1343.1	Kgm^2
l_f, l_r	Distance of wheel from CG along car axis (front, rear)	10.40, 15.60	m
C_{af}, C_{ar}	Tire Cornering Stiffness (front, rear)	976.5971	-- (unitless)
v_x	Longitudinal velocity	10	m/s

Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

Section III: Nominal System Analysis

Here, nominal system values are substituted into the state space equations to perform some preliminary analysis of the system on MATLAB.

a) Nominal System Parameters:

Nominal values for all the variables used in the system are provided in (Table 1). These values have been taken from a Class B hatchback car.

```
A =
         x1        x2        x3        x4
   x1     0         1         0         0
   x2     0   -0.3519         0    -9.085
   x3     0         0         0         1
   x4     0    0.7562         0    -51.12

B =
         u1     u2
   x1     0      0
   x2  1.76   1.76
   x3     0      0
   x4 15.12  22.69

C =
       x1  x2  x3  x4
   y1   1   0   0   0
   y2   0   1   0   0
   y3   0   0   1   0
   y4   0   0   0   1

D =
       u1  u2
   y1   0   0
   y2   0   0
   y3   0   0
   y4   0   0
```

```
vx =
    10
m =
        1110
Iz =
   1.3431e+03
lf =
   10.4000
lr =
   15.6000
g =
    9.8100
Fz =
   1.0889e+04
Fzf =
   5.4446e+03
Fzr =
   5.4446e+03
Caf =
  976.5971
Car =
  976.5971
```

b) Nominal Stability (Poles and Zeros)

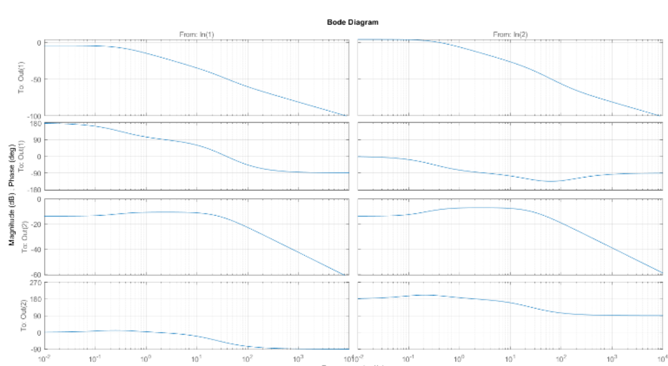
- Poles for the system are: $[-25.4168, -0.3189, -25.4168, -0.3189]$
- Zeros for the system are: $[-25.4168, -0.3189]$
- The system is nominally stable and has no RHP zeros.

c) Controllability and Observability

Using the $Q_c = [B,\ AB, A^2B \ldots]$ and $Q_o = [C; CA; CA^2 \ldots]^T$ matrices it was seen that the system is fully controllable and fully observable.

d) Frequency Analysis (Bode Plots)

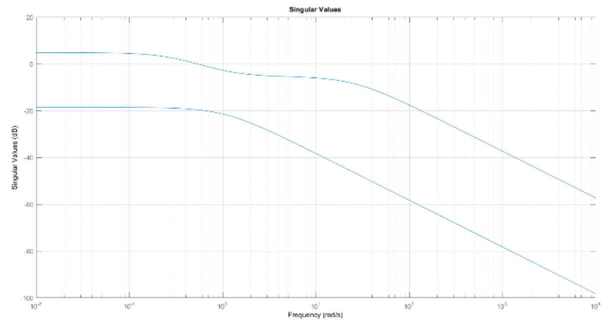
The system is stable as all the poles are on the left half of the imaginary axis. Since there are two inputs and two outputs, there are 2x2 bode plots.



Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

e) Singular Value Analysis

The singular value plot shows that the system has good low frequency command following and good disturbance attenuation properties at higher frequencies.



Section IV: Controller Design – Linear Quadratic Regulator Servo Control

The LQR is a controller which defines a cost function of the states and controls and attempts to minimize that cost by obtaining a suitable feedback gain matrix G.

A typical cost function (excluding state-control cross linking) is of the form:

$$J(u) \triangleq \int_0^{\infty} (x^T Q x + u^T R u)\, d\tau$$

To obtain the optimal control for the equation given above the Riccati equation is solved:

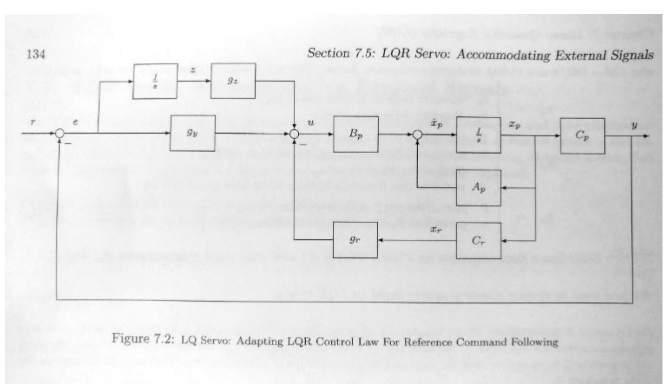
$$0 = KA + A^T K + M^T M - KBR^{-1}B^T K$$

By solving for K above, the feedback control gain matrix is obtained as:

$$G = R^{-1} B^T K$$

Finally, feedback control is provided as: $u = -Gx.$

As the name suggests, the LQR controller works well for regulation control problems. However, in order to track a reference signal a few modifications will need to be made to its structure. An integrator is augmented with the plant to allow for zero steady state error and the reference signal to be tracked is then injected into the system. This gives rise to the LQR Servo controller which can be represented as shown the block diagram below:



Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

The exact details for this project about the control gain matrix G, integrator state augmentation, reference signals, etc. is provided in Section VII Subsection A.

Section V: Controller Design – Model Predictive Control

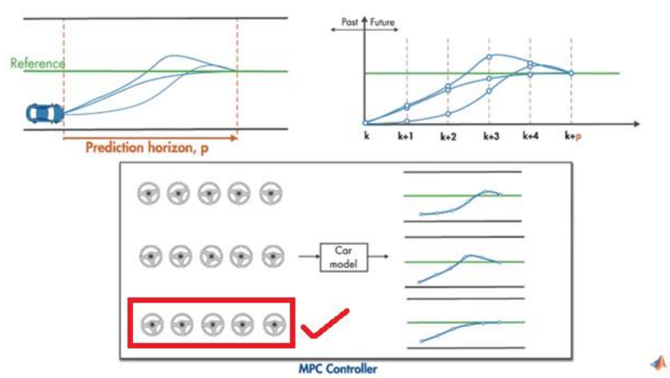
The MPC is a model-based controller. This means that similar to the LQR which requires the plant's system matrices/dynamics models, the MPC also requires a model of the systems dynamics.

The MPC also defines a cost function similar to the LQR and attempts to find the control inputs that would minimize the cost function.

$$Prediction\ Horizon = P$$

$$J = \sum_{i=1}^{P} w_e e_{k+i}^2 + \sum_{i=0}^{P-1} w_{\Delta u} \Delta u_{k+i}^2$$

The MPC goes a step beyond the LQR while minimizing the cost function, by making multiple prediction steps within a given time step of operation. Using the plant dynamics model provided to it and a '*prediction horizon*' (P) lengths, the MPC attempts to see how the system would respond to different control inputs over the given prediction horizon. In each step, for the trajectory that was taken during the prediction horizon's N steps, a cumulative cost is calculated and stored. This cost is compared against other system trajectories that are possible for a different set of control inputs. Finally, the set of control inputs that results in the least cost is then picked as the control to be provided to the system. This preview or look ahead that the MPC performs is beneficial in providing better overall tracking performance, as the controller can now consider the expected cost of taking an action not just at that moment in time but for the next few instants.



MPC also has a concept of the '*control horizon*'. This represents the length of control inputs that the MPC should actively try to optimize for. After optimizing for the desired number of control steps, the last control input calculated will be held for the rest of the prediction horizon, till the next optimal control sequence is calculated in the next MPC time step.

Another advantage that MPC has over the LQR is that it allows the output of the optimizer to be controlled using constraints placed on the states or control inputs of the system. Generally, hard constraints are placed on inputs to represent actuator constraints, while outputs have a soft constraint.

Finally, there are several types of MPC controllers that can be designed based on the linearity of the system, time available to perform online optimizations and depending on whether the plant is time varying or not. Each one has a different optimizer that is used internally to cater to the exact needs of that controller.

For this project, since the final state space equation was approximated mathematically to be a linear one, the vanilla, linear MPC has been used. More details about the implementation details is provided in Section VII Subsection B.

Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

Section VI: CarSim Co-Simulation

All of the above equations were tested first on MATLAB alone to validate their working. Once working and validated in MATLAB, their application was extended to a more realistic environment like CarSim.

Testing control algorithms on an actual automobile can be time consuming, labor intensive and expensive. To help resolve this issue simulations using physics backend engines to simulate real world dynamics helps quite drastically. CarSim is one such tool designed to test automobile systems in real-world like environment. It is a high-fidelity physics-and-controls application that allows controls applications to be controlled via MATLAB as a co-simulation.

- CarSim allows the user to define the physical properties of the car like mass, inertia, height, widths, wheel separation distance, center of gravity, etc.
- Over this, it also has several features that simulate the transmission, throttle, powertrain and gears that exist under the hood in cars.
- Other critical features like tire contact patch, tire cornering stiffness, suspension dynamics, can also be provided.
- The actual control of a car in CarSim is done quite similar to one in the real world. For example to make a car move forward, the brake needs to be released and throttle needs to be pushed down, while the gear must be in a non-neutral position. Based on the extent to which the throttle is pressed, the car's velocity will change.
- These control commands for throttle, steering, braking etc can be provided to the CarSim car using MATLAB and Simulink.
- For this project Class B Hatchback car has been selected in the software and the superior of the two control algorithms has been implemented for the car in two different road friction conditions to see how the controller's performance changes.

The exact parameters set and configuration changes made in CarSim are provided in "Section VII Results and Discussion".

Section VII: Results and Discussion

This section summarizes the experimental setup, simulation process and results obtained with each of the controllers. It consist of two parts. "Part 1 MATLAB" discusses the results obtained in the pure MATLAB environment, while "Part 2 MATLAB-CarSim Co-simulation" talks about the MATALB-CarSim co-simulated results.

- All the tests carried out were for a longitudinal velocity of $10\ m/s\ =\ 36\ kmph$.
- As this is considered low speed at which improved cornering ability is preferred, the front and rear steering wheels are expected to be moved in opposite directions.

PART 1: MATLAB Implementation

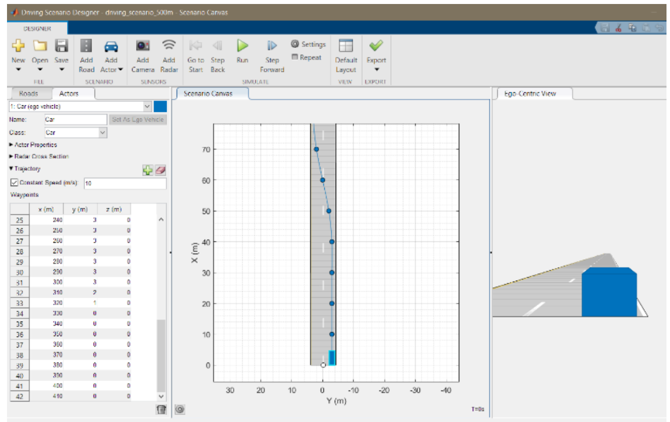
Before directly jumping into the CarSim co-simulation environment, I wanted to validate the working of both algorithms proposed, since it was the first time that I was understanding and analyzing how they perform. This section of the results is dedicated to this first step, where all the simulation was performed in MATLAB alone.

Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

A. Creation of Reference Signal

In this project the reference signal for this system is a desired lateral position and desired yaw angle for the automobile. When implemented in a MATLAB-only environment, these values are provided as waypoints that the vehicle should reach at given instants of time.

The lateral position and yaw angle are coupled inputs which depend on the path the vehicle had taken in the previous time step, and the path that it will be taking in the next time step. They are not completely independent and depends on the geometry of the path (spline) that is constructed between waypoints. To help with the spline extrapolation between waypoints, MATLAB's Driving Scenario Designer was used.



Here, various waypoints can be provided to the system, and based on the path that MATLAB creates between those waypoints, it outputs the reference signal of desired lateral position and desired yaw angle.

- The road constructed was 500 meters long, of which around 350 meters was covered by 40 waypoints.
- There were two lanes, each of 4m width.

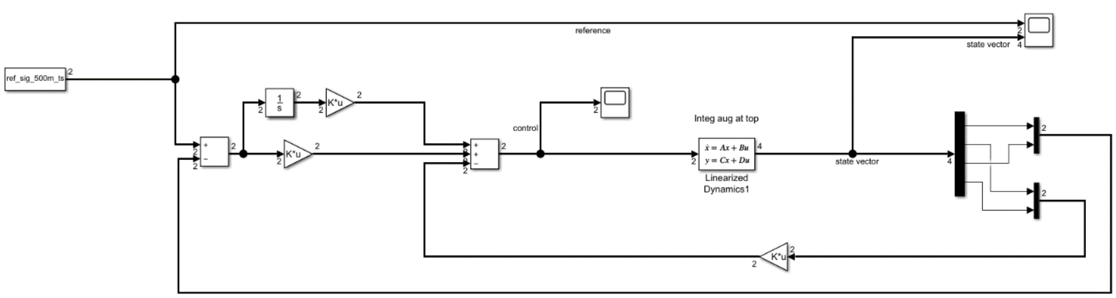
Note: This trajectory generation is possible via means other than a built-in MATLAB toolbox by using techniques like spline extrapolation, jerk minimizing quintic polynomial trajectories, etc. However, to keep the project more focused on the controls aspect than trajectory generation, the MATLAB Driving Scenario was used in the manner described earlier.

B. Linear Quadratic Regulator Servo Control

As explained in the earlier section, the LQR Servo system was setup on Simulink to allow for command following performance.

Controller Setup:

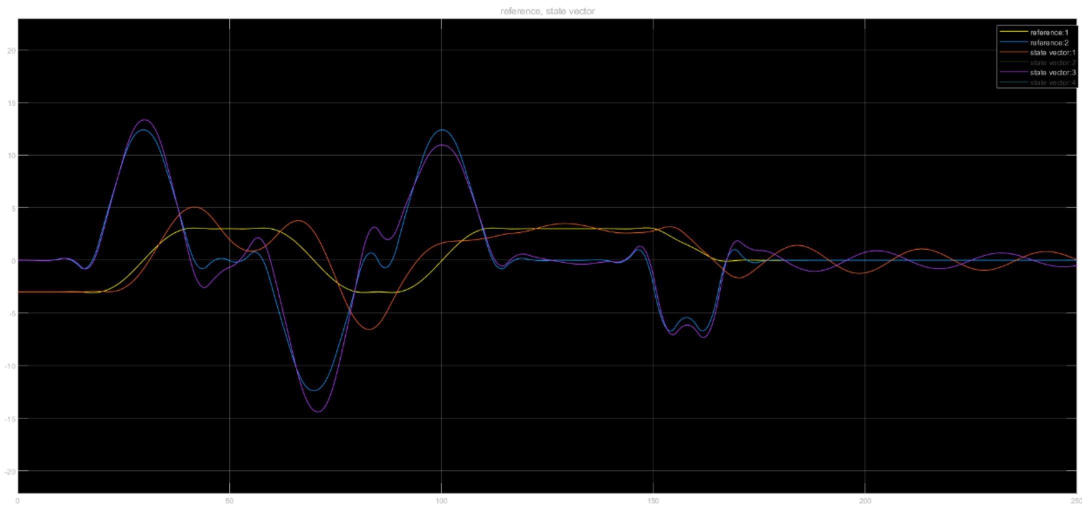
- A 2x2 integrator was augmented to the system so that each channel of the reference signal (having two channels) that needed to be tracked would have zero steady state error.
- Then, the LQR gain matrix G was calculated for the augmented system.
- Finally, G was split into three portions, with G_z being used with $x_z = [z1, z2]$ for the augmented integrator states, $G_y$was used with $x_y = [y, \psi]$ and $G_r$was used for $x_r = [\dot{y}, \dot{\psi}]$ the remaining two states of the system which were not necessary at the output for tracking.



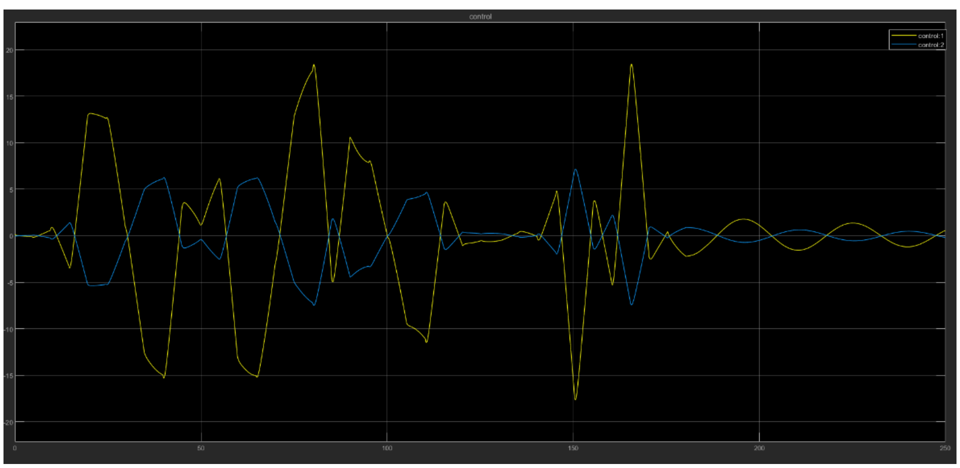
Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

Tuning Procedure:

- Initially, I started with $Q = diag(1,1,1,1,1,1)$ and $R = diag(0.1, 0.1)$ which did not yield satisfactory results at all.
- Later, I removed the weighting functions from Q that corresponded to the integrators and the response of the system seemed to improve, but was still very sluggish.
- Finally, I increased the weighting cost/importance of the error corresponding to lateral position and yaw rate, and also decreased R to $diag(0.001, 0.001)$ to get the results shown below.



Reference v/s Outputs: Lateral Position and Yaw Angle



Control Inputs: Front Steering angle del_f, Rear Steering angle del_r

Result:

- It can be seen from the above plots that the outputs track the reference signal.
- Signal values stayed within 15% – 20% of the reference value throughout the 250 seconds of simulation.
- By looking at the control input plots, we see that the front steering and rear steering are usually turning in a direction opposite each other, as desired at low speeds for faster cornering.

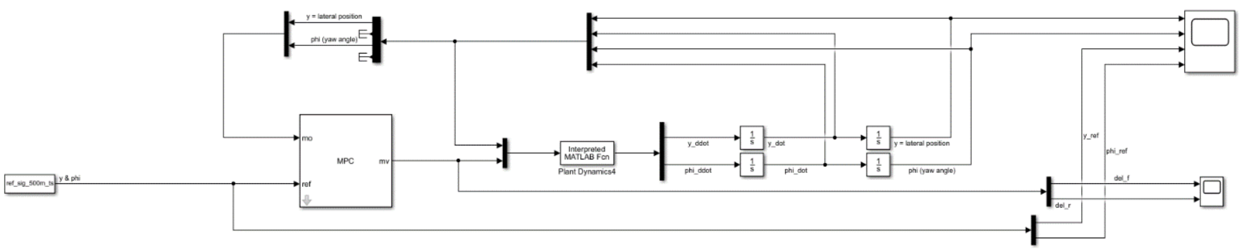
Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

C. Model Predictive Control

In the problem formulation, a linear plant model was used, hence a simple linear MPC controller was implemented to control it.

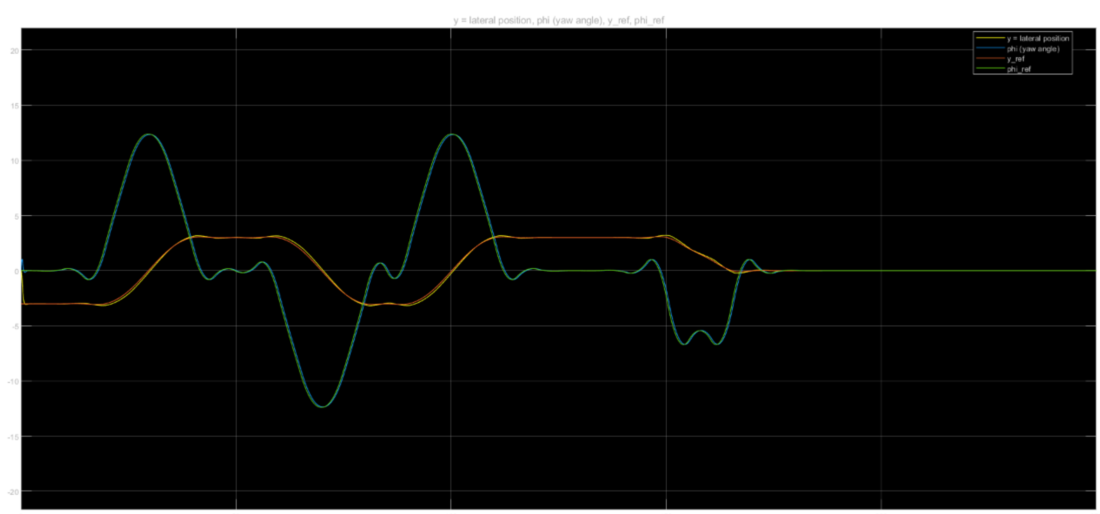
Controller Setup:

- The MPC controller was provided with an input being the reference signal (ref) that needed to be followed by the system.
- The measured output (mo) signals were also provided to the MPC controller as inputs.
- The dynamics equations were accessible by the MPC (via global variables) to predict the next state of the system in each of its online optimization steps.



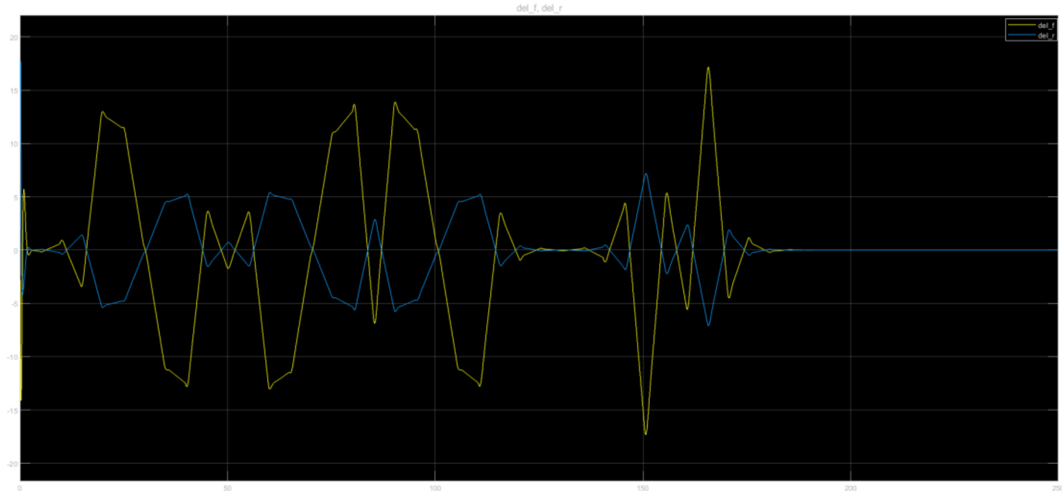
Tuning Procedure:

- Sample time for the MPC was set to 0.1 seconds, which was 10x the simulation time step of 0.01 sec.
- MPC was initialized with a prediction horizon of 3 and control horizon of 1. This did not give good step and ramp response (which is what lateral position and yaw angle reference signals are similar to). So, prediction horizon was increased to 10 and the control horizon to 5. Response improved drastically for this. A further increment in both resulted in slightly better performance, so the prediction horizon was set at 15 was the control horizon at 7.
- Hard input constraints were set to ensure steering angle remained between +/- 30 deg and steering rate was between +/- 15 deg/s.
- Finally weights in the cost function were adjusted from the default value of 1 for the state error and 1 for control increment to 0.382 and 0.552 respectively for each state and each control variable.



Reference v/s Outputs: Lateral Position and Yaw Angle

Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116



Control Inputs: Front Steering angle del_f, Rear Steering angle del_r

Results:

- Very good tracking performance was observed for both the lateral position and the yaw angle.
- Signal values stayed within 2% of the reference value throughout the 250 seconds of simulation.

D. Comparison Between LQR and MPC:

- Tracking error with the MPC controller was much lower than the LQR controller.
- Control effort for the MPC was also slightly lesser than the LQR controller.
- Both the above phenomena are likely due to the fact that MPC is able to predict the future states of the system during the prediction horizon to chose a more appropriate control sequence than just an abrupt value at the given point in time. It could also be that the internal optimizer being used by MPC is able to find better optimal values for this paradigm and the given tuning values above.

PART 2: MATLAB-CarSim Co-Simulation:

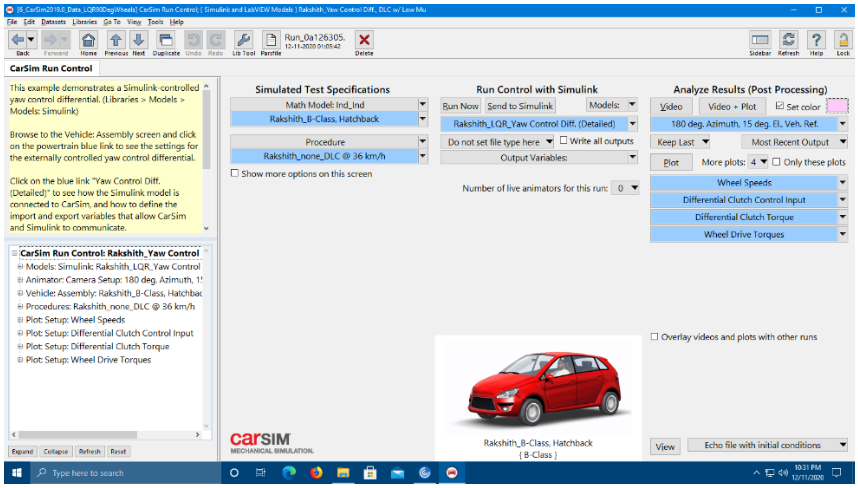
E. CarSim Co-Simulation

The purpose of this section of the project is to test both the controllers in a more realistic environment like CarSim. CarSim and its physics engine simulates many dynamics that were not modelled mathematically like suspension forces, powertrain delay, etc. It was expected that these unmodelled dynamics would result in poorer performance of the controllers in CarSim and would require further tuning.

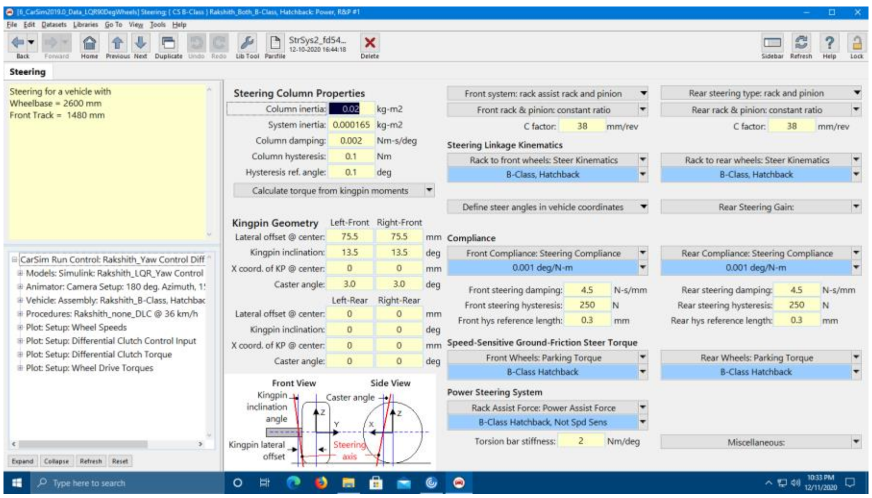
CarSim Setup:

- The Double Lane Change dataset was used as a template over which modifications were made as necessary for this project.
- The tire, powertrain, brakes, suspension were all kept unchanged from the template.
- In the steering settings rear steering was added for the car to make it four wheel steer capable.

Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116



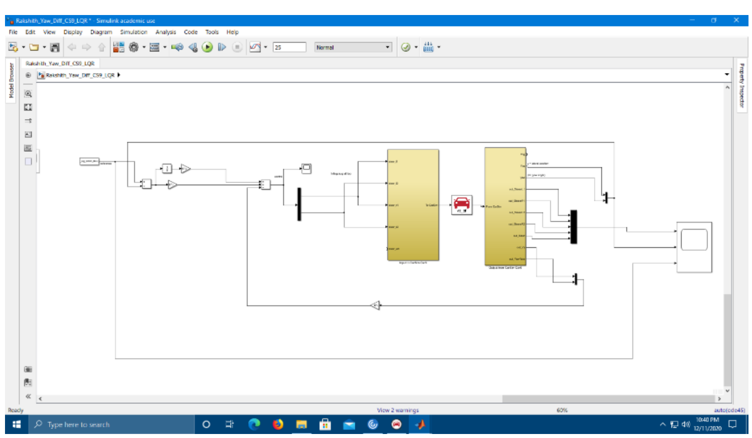
Home page displaying Double Lane Change home page with modified longitudinal speed, modified Simulink model file.



Steering page showing added rear steering for the car

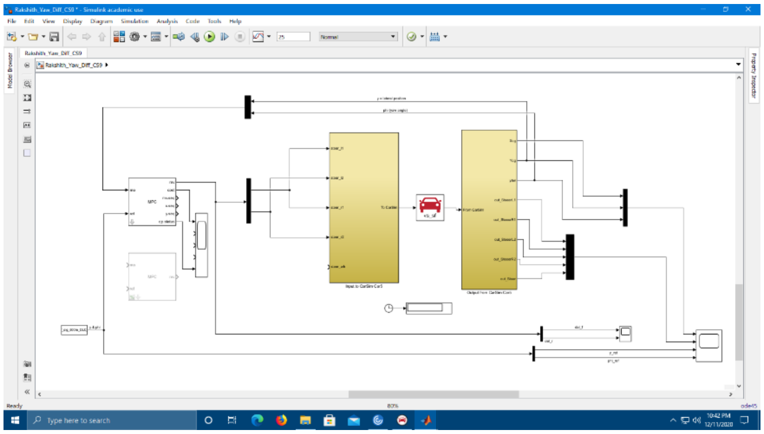
Controller Setup:

- For both controllers, the same exact block diagram was maintained in Simulink as show in earlier sections. However, the block representing the system dynamics was replaced with the CarSim s-block.
- LQR Servo block diagram:



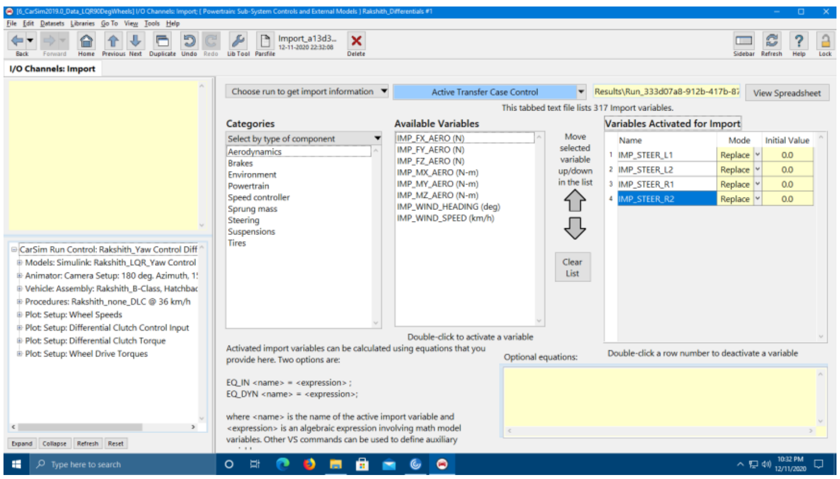
Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

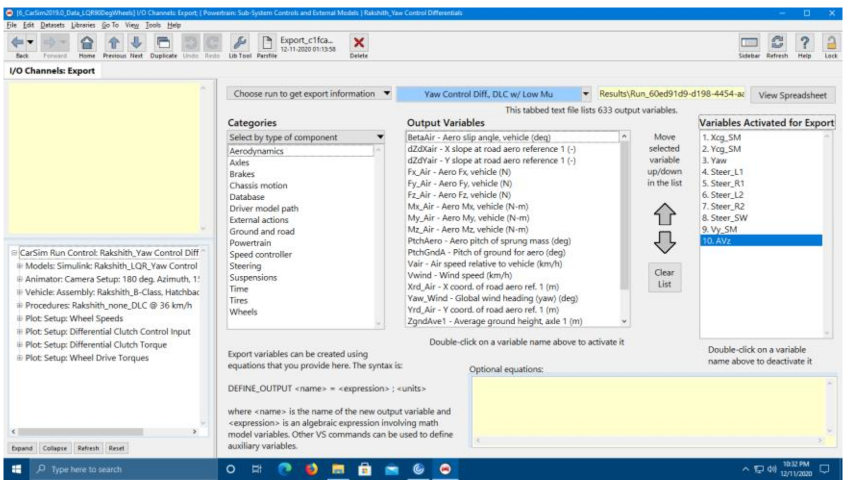
- MPC block diagram:



Simulink block with the CarSim s-function representing the vehicle in CarSim

- The inputs to the system were the front and rear wheel angles (left and right wheels were given the same angle value)
- The outputs from the model were the lateral position and yaw angle. In the case of the LQR servo, the lateral velocity and yaw angle rate were also taken as outputs.





Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

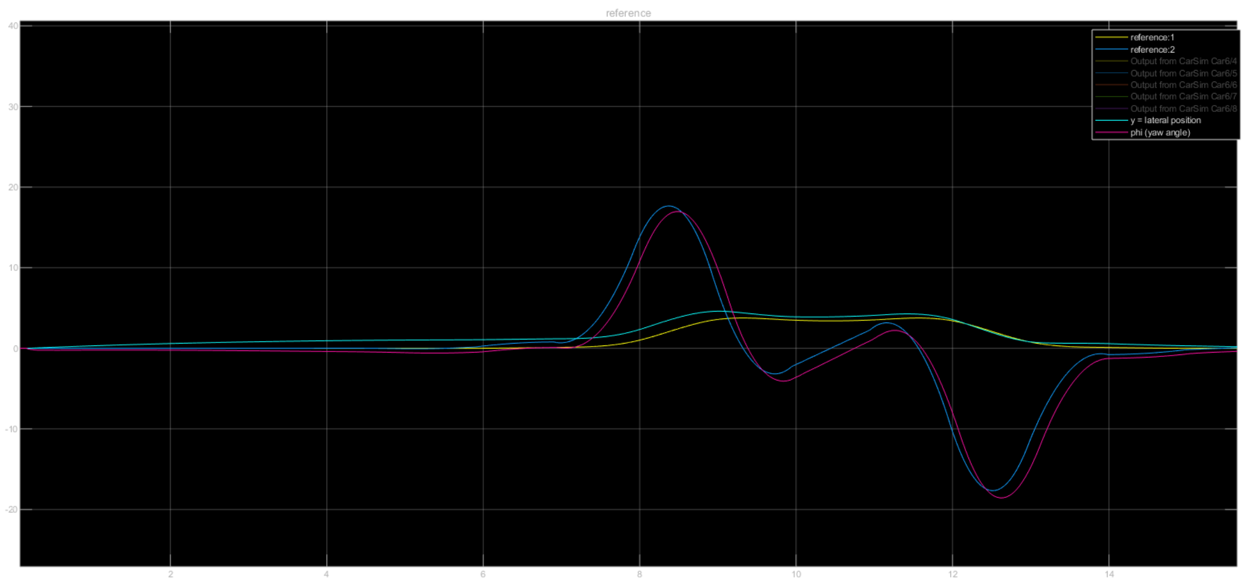
Tuning Procedure and Results for LQR Servo:

- Surprisingly, the same exact gain values calculated from the pure MATLAB simulation worked perfectly well for the LQR controller when implemented in CarSim as well.
- So, no additional tuning was required. Just a Simulink reconnection of blocks with the carsim s-function was needed.

```
>> G

G =

    6.6812    5.9761    1.4484    3.4892   31.6095    2.2505
    5.9761   18.4393  -49.9790  -55.0885    0.9161   12.5538
```

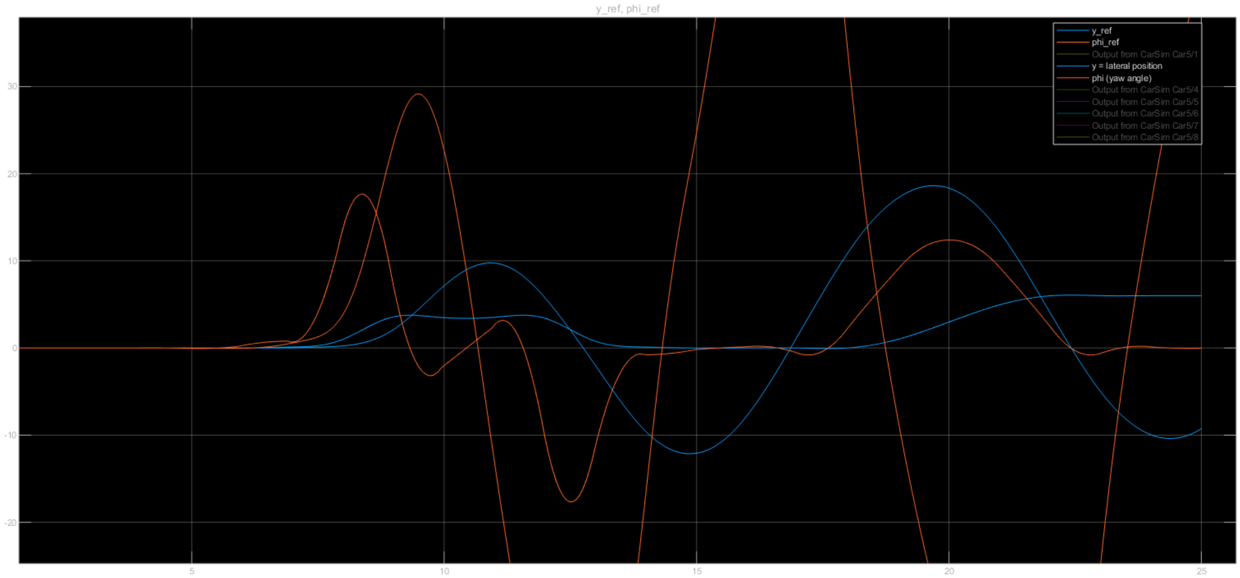


CarSim results for LQR Servo

Tuning Procedure and Results for MPC:

- The results for the MPC did not translate well from MATLAB to CarSim however.
- After tuning the input constraints, output constraints, cost function weights and cost function scaling values in several combinations, poor tracking results were obtained.
- Lateral position is reasonably close to the reference value, however due to the yaw angle not tracking the state well, the overall performance of lateral position and yaw angle tracking is bad.
- Further analysis is needed to identify why the yaw angle does not track the signal but instead slowly oscillated to infinity.

Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116



CarSim results for MPC

Section VIII: Conclusion and Future Work

The project allowed me to get a good understanding of car lateral dynamics and the effect of the added rear steering. It also helped me apply the concepts of LQR and MPC to the systems and observe how they performed in a pure MATLAB environment versus a more physics based CarSim environment.

There are still areas that can be worked on for further understanding and analysis:

1. The MPC controller that was implemented in CarSim needs to be analysed further to identify the reason for failed tracking. Compared to LQR, MPC is generally more robust since it has the ability to predict a few steps into the future using the plant model, however when implemented in CarSim this did not hold for some reason.
    a. A potential reason might be CarSim makes the plant non-linear in nature, so the linear MPC model created and tuned in MATLAB fails.

2. The plant that has been provided is time invariant only if the longitudinal velocity of the car is constant. This is a severe restriction in the real world. So as a next step, more suitable controllers like adaptive or gain scheduled controllers can be explored.

3. A major driving factor to use rear steering is to ensure side slip angle of the car is reduced for better steering capability. The side slip angle β can be introduced as another state of the system and a controller to drive β to zero can be explored.

Rakshith Vishwanatha
ASU ID: 1217756241
Mobile: 480-781-6116

References:

[1] "Optimal Model Following Control of Four-wheel Active Steering Vehicle", Proceedings of the 2009 IEEE International Conference on Information and Automation June 22 -25, 2009, Zhuhai/Macau, China

[2] "Predictive Active Steering Control for Autonomous Vehicle Systems", IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, VOL. 15, NO. 3, MAY 2007

[3] "LQR optimal control research for four-wheel steering forklift based-on state feedback", Journal of Mechanical Science and Technology, June 2018, 32(6):2789-2801

[4] "Sliding-mode control of four wheel steering systems", 2017 IEEE International Conference on Mechatronics and Automation (ICMA), Takamatsu, Japan

[5] Nourhazimi H. et. al. "Yaw Stability Improvement for Four-Wheel Active Steering Vehicle using Sliding Mode Control", 8th International Colloquium for Signal Processing and its Applications, 2012

[6] P Falcone, et. al. "A Real-Time Model Predictive Control Approach for Autonomous Active Steering", IFAC, 2006

[7] Rajamani, Rajesh. "Lateral vehicle dynamics." Vehicle Dynamics and control. Springer, Boston, MA, 2012. 15-46.

[8] A. A. Rodriguez. "Multivariate Feedback Control Systems", Control3D.

[9] Feng DU et al. "Robust Control Study for Four-Wheel Active Steering Vehicle", International Conference on Electrical and Control Engineering, 2010