Autonomous Vehicle Drifting with MPC

Abstract—This report describes a simulation for a vehicle drifting in a circle. The simulation considers vehicle dynamics, tire force dynamics, global position tracking, and drifting equilibrium conditions. The only input considered is the vehicle's steering angle with some constraints imposed on the range of steering inputs allowed and the rate at which it may change. The simulated vehicle could successfully track the circle trajectory in normal driving (non-drift) conditions. However, the simulation was not satisfactory under drift conditions. After much investigating, the conclusion for this outcome is an inappropriate drift condition reference. Future work should re-evaluate how this drift condition reference is designed.

Index Terms—Model Predictive Control, vehicle dynamics, drifting.

1 Introduction

Drifting is a term used by the motorsport community to describe controlled driving with rear tire saturation. The driver intentionally oversteers to initiate rear tire saturation, and sustains the saturation while controlling the vehicle trajectory. Professional drifting is a growing category of motorsports in which drivers display aggressive slip angles and yaw rate, while driving at high speeds.

Fig. 1: A Custom Nissan Sports Car Drifting

Simulating vehicle drifting requires consideration of car dynamics, tire force models, inputs such as steering angle and measurement of variables like velocity, the vehicle's global position, and yaw rate among others. This system can be characterized as a non-linear and multiple-input-multiple-output (MIMO) system. Given these two characteristics, Model Predictive Control (MPC) is ideally suited for this application.

2 Implementation Process

The project was conducted in three phases. Phase one established a working MPC in which a kinematic bicycle was modeled with an objective to track a circle. Phase two introduced vehicle dynamics and a tire force model for the same objective of tracking a circle. Phase three retained the vehicle dynamics and tire force model, and introduces the concept of drifting equilibrium to create a reference for drifting in a circle.

3 Model

3.1 Vehicle Parameters

The vehicle is modeled as a two-tired (front and back) bicycle model as suggested by Rajamani for simulation purposes. Dynamic equations are shown in equations 1 to 5 and its derivations can be found in Rajamani's paper [1].
Vehicle physical characteristics such as: distance from the center of mass (CoM) to the front of the vehicle (l_f), the distance from CoM to the rear of the vehicle (l_r), vehicle mass, and yaw inertia (I_z) will be used in the dynamic equations along with tire parameters for forces generated. The x velocity (v_x) of the vehicle was fixed. However, we experimented with a range.

- mass = 2000kg
- I_z = 4000 kg m^2
- l_f = 1.5m
- l_r = 1.5m
- v_x = 8 - 12.5 m/s

3.2 Vehicle Dynamics

There are five state variables that will be obtained from the dynamic equations: global X position, global Y position, yaw angle ψ, vehicle velocity ν_y, and yaw rate $\dot{r}$. These states are shown in figure 2

$$\dot{X} = v_x cos(\psi) - v_y sin(\phi) \tag{1}$$

$$\dot{Y} = v_x sin(\psi) + v_y cos(\phi) \tag{2}$$

$$\dot{\psi} = r \tag{3}$$

$$\dot{v_y} = -v_x r + \frac{1}{m}(F_{yf} cos(\delta) + F_{yr}) \tag{4}$$

$$\dot{r} = \frac{1}{I_z}(\alpha F_{yf} cos(\delta) - bF_{yr}) \tag{5}$$

3.3 Tire Dynamics

Modeling tires is the considered the most important component of an vehicle simulation [3]. Tires generate the forces that drive and maneuver a vehicle. Understanding the magnitude, direction and limit of tire forces is essential for vehicle control [6].

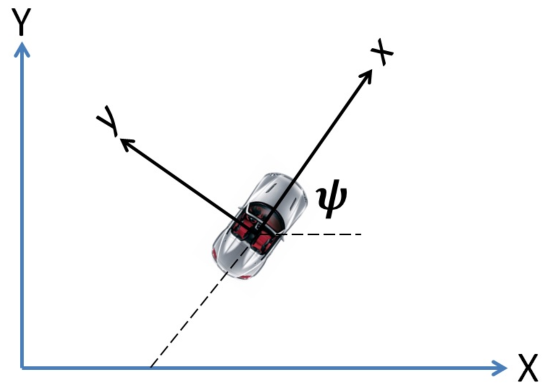

Fig. 2: Coordinate system used to define the state variables

When a vehicle is cornering, the lateral tire forces are a function of tire slip angle. Tire slip angle is the difference between the direction the tire contact patch is rolling vs. the direction the tire actually pointing. Slip angles occur due to the elastic properties of the rubber tires which deform under load.

Lateral tire forces vary linearly with slip angle for a certain range, which is typically minus four to plus four degrees [6]. After this point, tire forces vary non-linearly until it reaches a maximum - this is know as the tire saturation.

To model this behavior a reduced pacejka tire model is used. The inputs to the pacejka model are experimentally determined parameters. The advantage of the pacejka model is that is is easy to understand and use and its is commonly used in the high-end racing simulation industry. The disadvantage, however, is that the pacejka coefficients are generally proprietary data and so it is nor normally available to the public. John Gonzales from the UC Berkeley (UCB) MPC Lab kindly supplied experimentally determined lateral tire vs. slip angle data for this project.

Lateral tire force is given by the following equation as a function of slip angle. For the front tire the pacejka coefficients are b = 12.5, c=1.4, d=9810. For the rear tires, the coefficients are b=7.18, c=1.7, d = 9810.

$$F_y = dsin(ctan^{-1}(b\alpha) \tag{6}$$

Slip angle is calculated for the front and rear wheels.

$$\alpha_F = tan^{-1}(\frac{V_y + l_f \dot{r}}{V_x}) - \delta \tag{7}$$

$$\alpha_R = tan^{-1}(\frac{V_y + l_f \dot{r}}{V_x}) \tag{8}$$

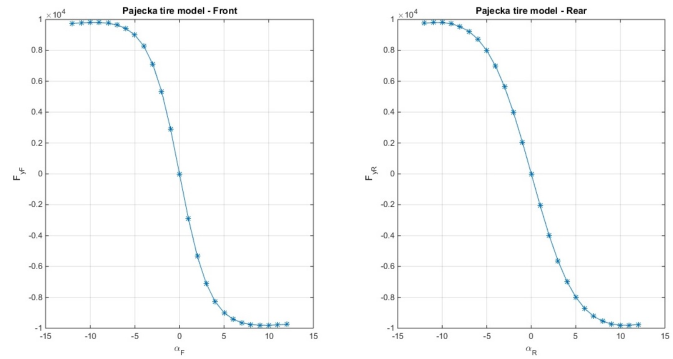

Fig. 3: Tire Force vs. Slip Angle for UCB Hyundai Test Vehicle

4 Model Predictive Control

A vehicle that tries to track a certain path can be considered a dynamic optimization problem where a certain group of equations govern the process. The general process can be thought of as:

$$x(k+1) = g(x(k)), u(k)), x(0) = x_0 \tag{9}$$

Equation 10 shows how the x values are updated based on a combination of previous x values along with a variable u. Normally the x variable is called the state variable and u is the input variable. The x and u are column vectors of any size. The updating nature of these equations is what creates the basis for an algorithm with an initial point or condition defined as x(0)=xo. It is common to set up these equations in the following mode called state space representation:

$$X(k+1) = Ax(k) + Bu(k) \tag{10}$$

For our scenario, the vehicle dynamic equations can not be expressed in a state space representation because they are not linear and can not be linearized easily; however our equations can be transferred from continuous time domain to discrete time domain and implemented in a non-linear solver. We achieve this through the Euler discretization method and the results are as follows:

$$X(k+1) = X(K) + (v_x cos(\psi(k)) - v_y sin(\phi(k)))\Delta t \tag{11}$$

$$Y(k+1) = Y(K) + (v_x sin(\psi(k)) + v_y cos(\phi(k)))\Delta t \tag{12}$$

$$\psi(k+1) = \psi(k) + r\Delta t \tag{13}$$

$$v_y(k+1) = v_y(k) + (\frac{1}{m}(F_{yf}(k)cos(\delta)(k) + F_{yr}(k) - v_x r(k)))\Delta t \tag{14}$$

$$r(k+1) = r(k) + (\frac{1}{I_z}(\alpha F_{yf}(k)cos(\delta(k)) - bF_{yr}(k)))\Delta t \tag{15}$$

The MPC algorithm structure is based on the key concepts of horizon and optimization. The horizon is the number of steps (this can be thought of as what lies ahead) that we can see forward and the optimization is done via the setup of a cost function that minimizes a set of values.

The cost function that we implement minimizes the difference between state values (usually X and Y for tracking and ν_y and yaw rate for drifting) and the reference values. The optimizer is implemented using a non-linear solver.

The simulation was implemented in Matlab using YALMIP. YALMIP is a modeling language that solves optimization problems. Within YALMIP, the IPOPT solver was used to handle the non-linear dynamics. The algorithm structure is shown below:

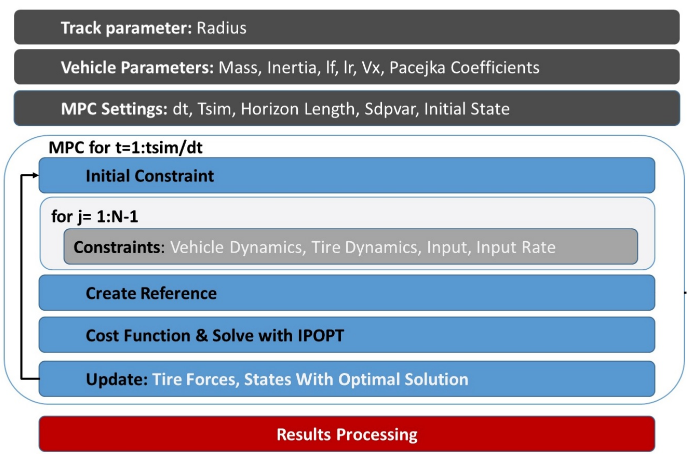

Fig. 4: MPC Algorithm Structure

We can see in the previous figure that the MPC algorithm starts by defining the physical parameters of the vehicle and its discretized dynamic equations. The horizon is defined along with a simulation time defined by its step size that was chosen at 200ms. The optimization is done with only the values of the horizon and only the first value of the optimized solution is fed back into the dynamic equations for updating and the process is repeated until the end of simulation time. Once simulation time is up, results are plotted and animations are rendered.

4.1 The Global Position Reference

The Global Position Reference (GPR) is created at each simulation timestep using. It tells the vehicle where it should be over the next N horizon steps. First, the GPR finds the closest point from the vehicle's Global X,Y position to the 30m radius circle. Second, from that point on the circle, the GPR calculates the next N horizon steps along the 30m radius circle as function of the vehicle's current velocity. The spacing between each of the reference steps is based on the vehicle's current velocity.

$$X_{circle} = \frac{X_{vehicle}}{\sqrt{X_{vehicle}^2 + Y_{vehicle}^2}} R \tag{16}$$

$$Y_{circle} = \frac{Y_{vehicle}}{\sqrt{X_{vehicle}^2 + Y_{vehicle}^2}} R \tag{17}$$

4.2 The Drift Condition Reference

The Drift Condition Reference (DCR) comprises y velocity, and yaw rate. It is obtained as a sequence of drifting equilibrium points that satisfy steady state conditions for the state variables of y velocity and yaw rate.

Therefore we can take equations 4 and 5 and set them to zero and proceed to solve for yaw rate and y velocity, as a function of a given steering angle.

$$0 = \frac{1}{I_z}(aF_y^{eq} cos(\delta^{eq}) + bF_R^{eq}) \tag{18}$$

$$0 = -v_x r^{eq} + \frac{1}{m} F_y^{eq} cos(\delta^{eq}) + F_R^{eq} \tag{19}$$

Figure 5 plots the result from solving the above equations. There are three sets of equilibrium conditions for both y velocity and yaw angle, corresponding to normal corning, drifting in a clockwise direction, and drifting in a counter clockwise direction. Since the vehicle simulated in this project is driving in a counter-clockwise direction, only the equilibrium condition is marked in blue is extracted. The DCR is implemented to generate an N horizon reference as follows:

- For the previous steering angle input, evaluate the set of beta and yaw angle equilibrium points.
- Using beta, calculate y velocity since x velocity is known (and fixed in this project).
- Fix the equilibrium y velocity and yaw rate for the remaining steps in the horizon to generate the DCR.

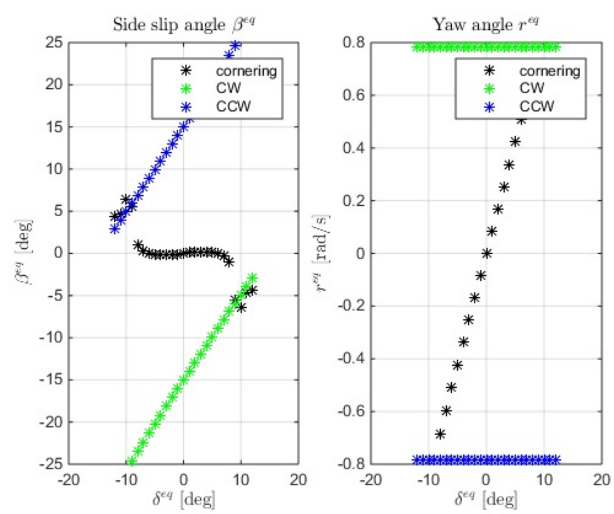

Fig. 5: Equilibrium Condition

4.3 Cost Function

For position tracking, the cost function is simply a minimization of the difference between the vehicle's current Global X,Y position and the GPR reference over N horizon. The cost loops over N steps.

$$\min_{z,u} J = \left| \left| \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} - \begin{bmatrix} z_{ref,1} \\ z_{ref,2} \end{bmatrix} \right| \right|^2 \tag{20}$$

Similarly, when simulating drift conditions, the cost function was modified to reflect the v velocity and yaw rate.

$$\min_{z,u} J = \left|\left| \begin{bmatrix} z_4 \\ z_5 \end{bmatrix} - \begin{bmatrix} z_{ref,4} \\ z_{ref,5} \end{bmatrix} \right|\right|^2 \quad (21)$$

5 Phase Implementations and Results

5.1 Phase 1: MPC and Kinematic Bicycle Model

Phase 1 is intended to apply the first basic principles of MPC with bicycle kinematic equations as shown below:

$$\dot{X} = v_x cos(\psi) - \beta \quad (22)$$

$$\dot{Y} = v_x sin(\psi) + \beta \quad (23)$$

$$\dot{V} = a \quad (24)$$

$$\dot{\psi} = \frac{v}{l_r} sin\beta \quad (25)$$

$$\dot{\beta} = \frac{l_r}{l_f + l_r}(tan(\delta_f)) \quad (26)$$

This set of equations do not account for tire forces unlike equations 1 to 5, therefore drifting is not possible; however, this is an excellent scenario for tracking and MPC implementation since the dynamic system is still non-linear and MIMO.

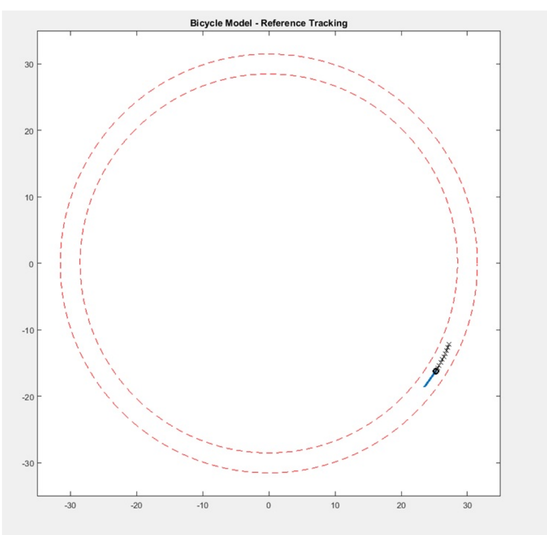

Fig. 6: Kinematic Bicycle Tracking a Circle With Open Loop Trajectory

The results were optimal and circle tracking was accurate. We can see results of the bicycle under the following conditions: a circle of 30 meters radius at a constant speed of 10-12 m/s and bicycle does approximately 1 lap.

5.2 Phase 2: MPC and Vehicle Dynamics

Given that Phase 1 yielded desired results, we proceeded to change the kinematic bicycle equations for the dynamic vehicle equations that account for forces. The MPC is again tested for tracking purposes only. The under the same conditions under Phase 1. Results show that the vehicle can track the circle reference even when more complex vehicle dynamics are included along with tire forces.

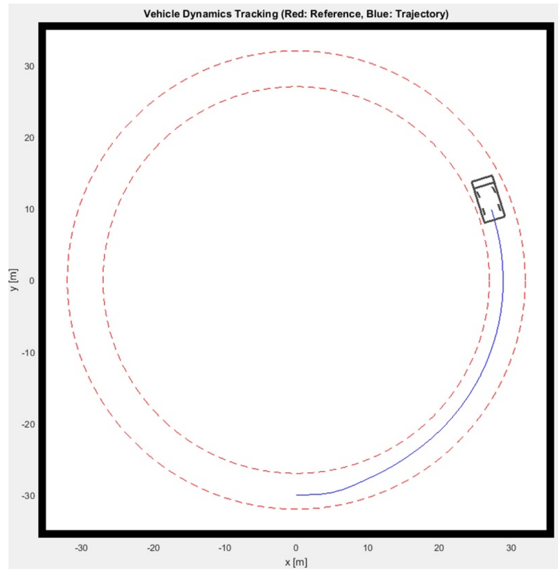

Fig. 7: Kinematic Vehicle Tracking a Circle

5.3 Phase 3: MPC and Drifting

Drift Dynamics were added in the form of Constraints and References. Figure 7 illustrates our best attempt at drifting. The vehicle slip angle is large, however, we are not able to sustain rear tire Saturation for a significant amount of time.

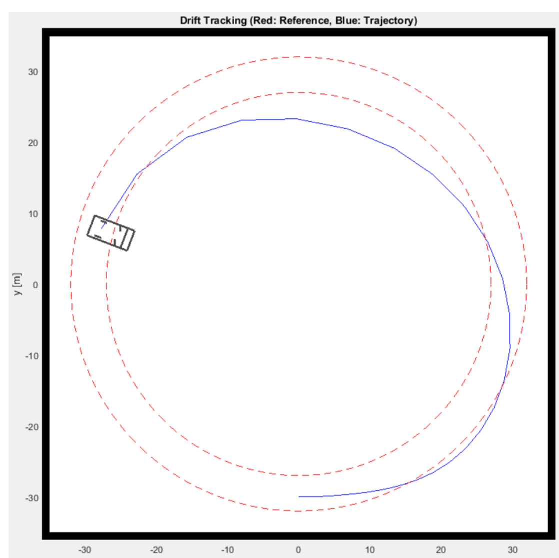

Fig. 8: Kinematic Vehicle Drifting Around a Circle

6 Results Summary

Implementation started with a bicycle model using MPC to track a circular trajectory. The results were aligned with expectations which allowed us to verify the MPC structure and proceed to use it for Phase 2. Phase 2 implemented non-linear Vehicle Dynamics within our MPC structure to track the circle. Results were also aligned with expectations. The use of more complex non-linear dynamics and tire forces made calculation time longer but proved that reference tracking is still aligned with expectations. Finally, with the Pacejka Tire Model implemented, we attempted to incorporate drift conditions into our reference trajectory which include rear tire saturation and a large. The results were unsatisfactory as rear tires didn't sustain rear tire saturation and therefore exhibited consistent β angle.

7 Conclusion and Future Work

Our project attempted to simulate a vehicle drifting around a circular trajectory. Our MPC algorithm was successful for tracking under simple bicycle kinematics and even vehicle dynamics. Our final step was to create a Drifting Reference based on drifting equilibrium and have MPC trace it. The results are not as clear as in the previous simulations, for instance we do not have a precise circle tracking but we do obtain moments where tire saturation is achieved but it is not fully maintained. We suspect this is due to the design of the Drifting Reference conditions and future work is to re-evaluate and design this.

Acknowledgments

The authors would like to thank Professor Francesco Borelli for his teaching and aid in developing our algorithm structure. Mr. Jon Gonzalez for his valuable help and Grace for his support and advice on the direction to follow.

References

[1] Rajamani, R. *Vehicle Dynamics and Control* 1st ed. Springer, U.S., 2006
[2] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.
[3] Hindiyeh, R.Y. *Dynamics and Control of Drifting in Automobiles*. PhD Dissertation Stanford University, 2013
[4] Gonzales, Jon. *Design of Figure Eight Drift Controller Using NMPC.*. University of California, Berkeley, 2015. Print
[5] Werling, M. and Reinisch, R. *Robust power-slide control for a production vehicle*. BMW Group Research and Technology, Int. J. Vehicle Autonomous Systems, Vol. 13, No. 1, 2015
[6] Erdogn, G. *Tire Modeling - Lateral and Longitudinal Tire Forces*. University of Minnesota, 2009. Retrieved from: http://www.menet.umn.edu/ gurkan/
[7] Whitehead, R. *A study of the properties that influence rollover propensity*. M.S. Thesis, Auburn University, 2005