Discrete Model Prodective Control on a Bicycle Dynamic Model

Apurva Sontakke,Nalin Bendapudi, Varun Shetty
{apurvaps, bnalin, vashetty}@umich.edu

***Abstract*—This report chronicles our the work done in EECS561-Digital Control Systems Project. The objective was to do trajectory optimization-based control of a car so that it smoothly traverses a given track. We have assumed that the dynamics of the car work according to the bicycle model. A proportional controller was used to obtain the initial trajectory of the car. The dynamic model was linearized around this trajectory, and a model predictive controller to find an optimized trajectory.**

***Index Terms*—Model predictive control, linearization, quadprog, bicycle model.**

I. Introduction

In this section, we'll talk about the dynamic model we used, our problem statement, and the motivation behind using MPC.

A. Problem Statement

Our team proposes to control an autonomous vehicle modeled by the bicycle model. The objective will be to track a pre-defined race-track whose Cartesian coordinates are known. This builds on the controls project of the Self Driving Cars course, which all three of us had taken in Fall 2019. For the 535 project, we had used a PID controller but here we plan on using the discrete time MPC controller.We are planning to generate a trajectory by using discrete time MPC given the initial states such that it lies between the left border and the right border of the track and reaches the specified end position.The lateral tracking error that we hope our controller will handle is +/- 2m.

Figure 1 shows the track we are using.

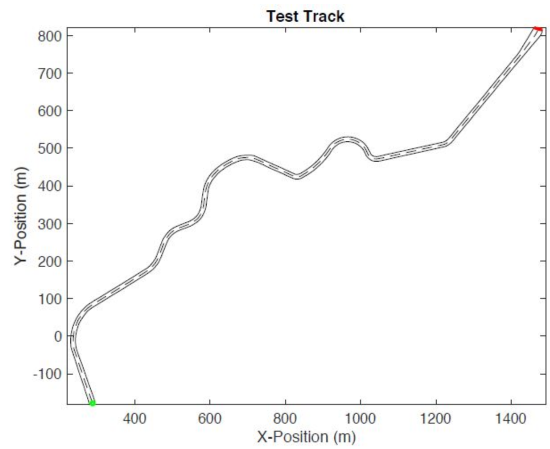

Fig. 1. Original Track

B. Bicycle Model

The non-linear bicycle model is given as follows:

$$\begin{bmatrix} \dot{X} \\ \dot{u} \\ \dot{Y} \\ \dot{v} \\ \dot{\psi} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} ucos\psi - vsin\psi \\ \frac{1}{m}(-fmg + N_w F_x - F_{yf}sin(\delta_f)) + vr \\ usin\psi + vcos\psi \\ \frac{1}{m}(F_{yf}cos(\delta_f) + F_{yr}) - ur \\ r \\ \frac{1}{I_z}(aF_{yf}cos(\delta_f) - bF_{yr}) \end{bmatrix} \tag{1}$$

The lateral forces F_{yf} and F_{yr} are described using Pacejka "Magic Formula"

$$F_{zf} = \frac{b}{a+b}mg \tag{2}$$

$$F_{yf} = F_{zf}D_y sin(C_y \tan^{-1}(B_y \phi_{yf})) + S_{vy} \tag{3}$$

$$F_{zr} = \frac{a}{a+b}mg \tag{4}$$

$$F_{yr} = F_{zr}D_y sin(C_y \tan^{-1}(B_y \phi_{yr})) + S_v y \tag{5}$$

where

$$\phi_{yf} = (1 - E_y)(\alpha_f + S) \tag{6}$$

$$\phi_{yr} = (1 - E_y)(\alpha_r + S_{hy}) + \frac{E_y}{B_y}\tan -1(B_y(\alpha_r + S_{hy})) \tag{7}$$

where α_f and α_r are the front and rear lateral slip angles which are given in degrees in the previous formulas. The front and rear lateral slip angles which is described in radians is given by:

$$\alpha_f = \delta_f - \tan^{-1}(\frac{v + ar}{u}) \tag{8}$$

$$\alpha_r = -\tan^{-1}(\frac{v - br}{u}) \tag{9}$$

Additionally, combined longitudinal and lateral loading of tires will be limited to F_x^* and F_{yr}^* in the following manner:

$$F_{total} = \sqrt{(N_w F_x)^2 + (F_{yr})^2} \tag{10}$$

$$F_{max} = 0.7mg \tag{11}$$

If $F_{total} > F_{max}$:

$$F^{x} = \frac{F_{max}}{F_{total}} F_x \tag{12}$$

$$F^{yr} = \frac{F_{max}}{F_{total}} F_{yr} \tag{13}$$

The inputs into this model are δ_f , which is the front wheel steering angle; and F_x, which is the traction force generated at each tire by the vehicle's motor. The vehicle begins from the following initial condition:

$$\begin{bmatrix} x \\ u \\ y \\ v \\ \psi \\ r \end{bmatrix} = \begin{bmatrix} 287[m] \\ 5[m/s] \\ -176[m] \\ 0[m/s] \\ 2[rad] \\ 0[rad/s] \end{bmatrix} \tag{14}$$

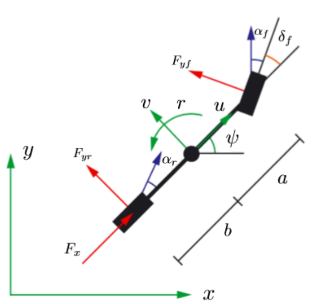

Fig. 2. An illustration of the bicycle model used to define the vehicle's dynamics.

II. Methodology

MPC (Model Predictive Control) is a time control strategy in which the future steps i.e the sequence of the control steps is determined my minimizing the cost function (objective function) at each time step over a horizon dependent upon the equations and constraints of the model.In short Model Predictive Control is used to predict the future behaviour of the system. One of the most important advantage of using MPC is that it can handel multiple input and multiple output systems and interactions between them.

TABLE I
Dynamic Model Constants

Vehicle Parameter	Value
δ	[-0.5,0.5]
F_x	[-5000,5000]
m	1400
N_w	2
f	0.01
I_x	2667
a	1.35
b	1.45
B_y	0.27
C_y	1.2
D_y	0.7
E_y	-1.6
S_{hy}	0
S_{vy}	0
g	9.806

We use the linear MPC because of its ability to find a global and optimal minimum by placing constraints in the inputs and states. We use the linear MPC algorithm where we apply successive linearization to develop a time-variant linear system.

We obtain a linear model of our dynamics by calculating an error model about the reference trajectory. In order to generate the reference trajectory we used a Proportional controller by calculating the lateral error between the center-line of the track and the current position of our vehicle. This error is used as the feedback reference to then generate the steering command.

A. Trajectory Tracking using MPC

While obtaining the nominal trajectory we set a constant lateral velocity and this leads to a reduced order model. This reduced order model is:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{v} \\ \dot{\psi} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} ucos\psi - vsin\psi \\ usin\psi + vcos\psi \\ \frac{1}{m}(F_{yf}cos(\delta_f) + F_{yr}) - ur \\ r \\ \frac{1}{I_z}(aF_{yf}cos(\delta_f) - bF_{yr}) \end{bmatrix} \tag{15}$$

We apply Euler integration to generate a discrete time system by approximating the dynamics as follows:

$$x(k+1) = x(k) + dt \times (A(k)x(k) + B(k)u(k)) \quad (16)$$

where $x(k) \in \mathbb{R}^n, A(k) \in \mathbb{R}^{n\times n}, B(k) \in \mathbb{R}^{n\times m}$. We discretized the system at a sampling time of 0.01s and used a prediction horizon of 10 steps. For our model, A(k) and B(k) are as follows:

$$A(k) = \begin{bmatrix} 0 & 0 & A_{13} & -sin(\psi_{ref}) & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \frac{-(Ca_r+Ca_f)}{mu_0} & \frac{(Ca_r b - Ca_f a - mu_0^2)}{mu_0} \\ 0 & 0 & 0 & \frac{(Ca_r b - aCa_f)}{I_z u_0} & \frac{-((b^2 Ca_r)+(a^2 Ca_f))}{I_z u_0} \end{bmatrix} \quad (17)$$

where $A_{13} = -u_0 sin(\psi_{ref}) - v_{ref} cos(\psi_{ref})$

$$B(k) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{Ca_f}{m} \\ \frac{aCa_f}{I_z} \end{bmatrix} \quad (18)$$

To find the optimized solution we use quadratic programming that is specified by:

$$min \frac{1}{2} x^T H x + f^T x \quad (19)$$

such that $Aeq * x = beq$ and $lb \leqslant x \leqslant ub$. H and Aeq are matrices and beq, lb abd ub are vectors.

The initial conditions of the dynamics are satisfied in the first three rows of Aeq and beq. All subsequent rows of beq are zero. Each subsequent row of Aeq represents the Euler integration steps in order from time step k to time step $k + 10$. Aeq encodes the linear equalities and is a $M \times N$ matrix where M are the number of equalities i.e. in our case 55 (number of states $\times$ number of state decision variables); N is the number of total decision variables. beq is a vector with M rows. An example of Aeq for the first time step is:

$$\begin{bmatrix} Aeq(6:10, 6:10) \\ Aeq(6:10, 1:5) \\ Aeq(6:10, 56) \end{bmatrix} = \begin{bmatrix} \mathrm{I}_5 \\ \mathrm{A}(2) \\ \mathrm{B}(2) \end{bmatrix} \quad (20)$$

The remaining elements of Aeq can be filled in a similar fashion. H is the quadratic objective term. We construct it using a quadratic penalty of $Q = [1, 1, 0.5, 1, 1]$ and $R = 0.1$.

We compute the states using ode45 and using the inputs generated from MPC.

III. Results and Discussion

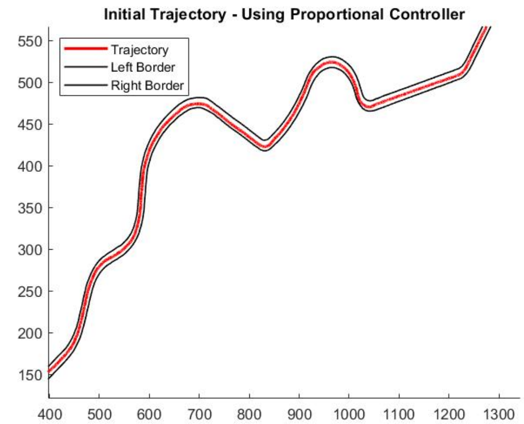

Fig. 3. Initial trajectory generated using Proportional Controller

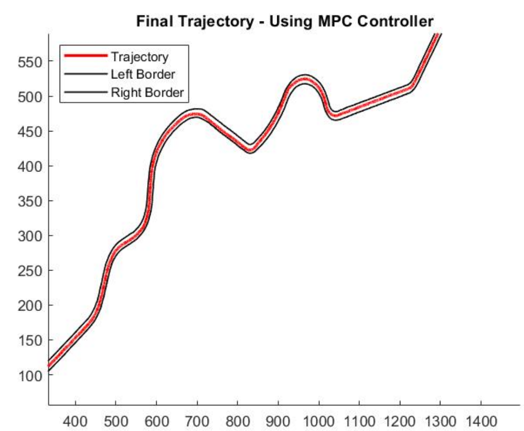

Fig. 4. Final trajectory generated using MPC

The MPC trajectory tracks the reference trajectory perfectly, even when the initial state is a little disturbed.

IV. Conclusion

The objective in the problem statement has been satisfied. The car modeled by the bicycle model tracks a reference trajectory even though the MPC controller used a reduced-order dynamic model, and initial state was altered. Some conditions under which the MPC controller wasn't giving good tracking performance was when initial position was altered too much ($> 2m$) or when an imperfect reference trajectory was provided. The code for the MPC controller is documented and open-sourced on GitHub.

V. Contribution

Apurva Sontakke	Code, Report
Nalin Bendapudi	Code, Report
Varun Shetty	Code, Report

VI. Appendix

The code for this project is uploaded on GitHub and can be accessed through this link.