Discrete MPC - Digital Controls Project Proposal

Apurva Sontakke, Nalin Bendapudi, Varun Shetty
April 21, 2020

1 Introduction

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. We would simulate the MPC controller with an initial state different from that given to the initial trajectory generator (PID controller). We expect that MPC will be able to track the reference trajectory even if initial state differs by $(\pm 2m, \pm 2m)$ and if there's a 5% random gaussian noise in the state estimate.

2 Model

The non-linear bicycle model is given as follows:

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

Here X and Y are the global cartesian coordinates, ψ is the yaw angle, u is the tangential velocity, v is the lateral velocity, and r is the angular velocity of the vehicle. The only control inputs are δ_f, the front wheel steering angle and F_x, the traction force generated at each tire by the vehicle's motor. The lateral forces F_{yf} and F_{yr} are described using the Pacejka "Magic Formula". The Non-Linear bicycle model is as shown in the figure 1.

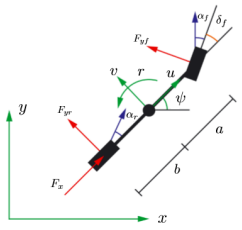

Figure 1: Illustration of the bicycle model used to define the vehicle's dynamics.

3 Implementation

We will implement a *discrete PD controller* (on the error defined as function of the deviation from the center-line of the race-track) to generate an approximate trajectory that our car will follow. We will use MATLAB's *ode45* solver to generate this trajectory. We will then use this trajectory to linearize our model and attempt to implement a *discrete MPC* controller to follow this trajectory. We will solve the MPC optimization problem using MATLAB's *quadprog*.