AA203 Final Project Report

Ricardo Paschoeto[1] and Srikanth Vidapanakal[2]

[1]rp304154@stanford.edu
[2]vsrikant@stanford.edu

June 6, 2021

Abstract

We present our study and conclusions using Model Predictive Control (MPC) in an autonomous vehicle model and integrated with a simulation environment (CARLA simulator). Our experiment focuses on imitation learning in the autonomous vehicle domain. We concentrate our efforts to bring the dynamics and costs from the results reached by YukunXia[2] (Carla, iLQR, and MPC approaches) to use in the Differentiate MPC library. We will discuss the results, acquired experience and analyze the two approaches.

1 Introduction

The project is an opportunity to apply the knowledge learned in the course in a fascinating area of an autonomous vehicle. This area evolves disciplines like Math, Physics, Mechanical Engineering, Computer Science, Electrical Engineer, and grows inside the industry and academic environment, generating opportunities in research and development.

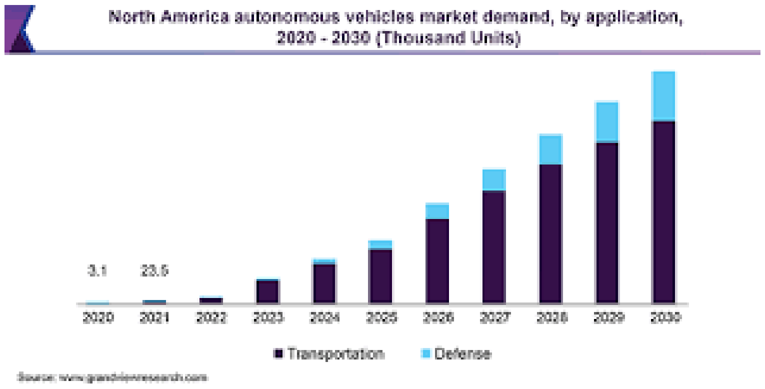

Figure 1: Growth Autonomous vehicle market

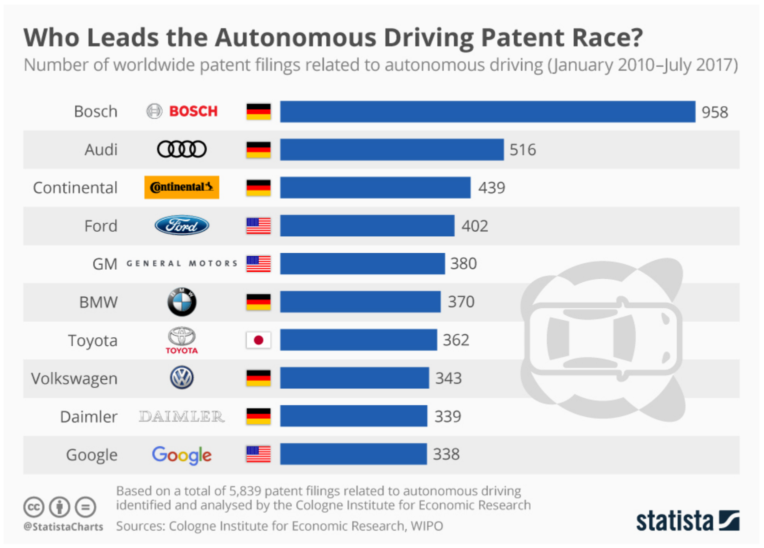

Figure 2: Patent filings

2 Related Work

In the last years, we have a considerable number of works related to Model Predictive Control (MPC), indirect methods, and iLQR to applying in autonomous systems. In our project, this material was essential to create the foundations.

Differentiable MPC for End-to-end Planning and Control [1], in this work, the authors presented the foundations of Differentiable iLQR. They used this concept to implement Differentiable Model Predictive Control (MPC) as a Python library.

Real-time MPC with iLQR for Self-Driving in Carla [2], In this work the authot presented the concepts to apply iLQR and MPC to an autonomous vehicle system using Neural Networks together explicit system dynamics.

Robust Model Predictive Control for Autonomous Vehicle/Self-Driving Cars [3], in this paper the authors presented an approach for controlling front steering of an autonomous vehicle. The paper was our source regarding to Bicycle Model and model dynamics.

Control-Limited Differential Dynamic Programming [4], the authors propose a generalization of DDP which box inequality constraints on the controls using indirect methods and iLQR. The approach proposed was applied in a Car Parking and Humanoid Robot.

Constrained Iterative LQR for On-Road Autonomous Driving Motion Planning [5], the authors presented a new implementation of the iLQR algorithm to an autonomous driving car but with constraints applied (CILQR).

End-to-end Driving via Conditional Imitation Learning [6], in this paper the authors presented a new approach to imitation learning policy, generate by on high-level command input. They showed a simulation using the CARLA simulator.

3 Problem Statement

The task of the project is an Autonomous vehicle simulation and, to reach this task, we integrated three main points:

1. A set of data to respect to control parameters from a Neural Network to calculate part of system dynamics
2. A vehicle environment that gave us the simulated measuring of car movement and the reference trajectory
3. A class Differentiable Model Predictive Control using Pytorch matrices and vectors with the cost function,

$$C = \frac{1}{2}\tau_t^T C_t \tau_t + \tau_t^T c_t$$
$$s.t.\ x_{t+1} = f\left(x_t, u_t\right)$$
$$x_0 = \ x_{init}$$
$$u_{lower} \ \leq u \leq u_{upper}.$$

4 Approach

We developed the project over three (3) phases:

Phase 01
We needed to know about the process presented in Differentiate MPC [1]. This process was time-consuming and focused on analyzing the math and algorithm Differentiable LQR [1], this algorithm was used as a start point of study to proceed with the proposed Differentiable MPC: Then we proceed to Differentiate MPC algorithm to solving the non-convex control problem:

$$x^*_{1:T}, u^*_{1:T} = argmin_{x_{1:T}\in X, u_{1:T}\in U} \sum_{t=1}^{T} C_t\left(x_t, u_t\right)$$

Subject to $x_{t+1} = f(x_t, u_t)$
$$x_1 = x_{init},$$

The code supports a quadratic cost function C and non-linear system transition dynamics$f$.

Phase 02
We looked for studies related to autonomous vehicles, mainly on the dynamics and costs functions, and these representations in a Pytorch code to adapt it to the Differentiate MPC class. The proposal presented in [2] was fascinating in all of these aspects, it gave us a good approximation

of Dynamics with neural networks (part of the dynamical is a black box, and could be represented by a NN) and the costs in Quadratic form,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{v} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} v\cos(\psi+\beta) \\ v\sin(\psi+\beta) \\ \frac{v}{l_r}\sin\beta \\ a \\ \tan^{-1}(\frac{l_r}{l_f+l_r}\tan(\delta_f)\ ) \end{bmatrix}$$

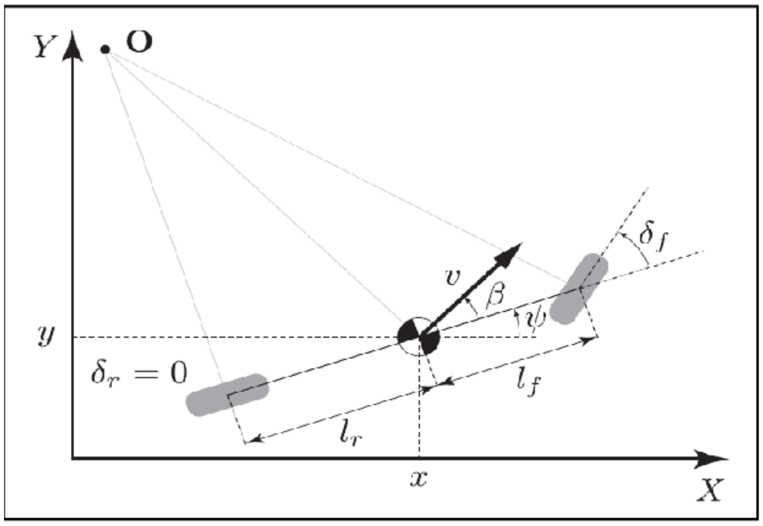

Figure 3: Bicycle Dynamics.

With final model has the following form,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{v} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} v\cos(\psi+\beta) \\ v\sin(\psi+\beta) \\ \frac{v}{l_r}\sin\beta \\ f_1(v,\beta,steering,\ throttle,\ brake) \\ f_2(v,\beta,steering,\ throttle,\ brake) \end{bmatrix}$$

Where$f_1,\ f_2$are outputs of NN.

1. $c_{final} = (x_f - x_{target})^2 + (y_f - y_{target})^2 + v_f^2$
2. $c_{running} = 0.001 * [steering^2 + throttle^2 + brake^2 + throttle * brake]$

The reference cost functions with heading/ cross track error:

1. $cost_{final} = \left[(x_f - x_{target})^2 + (y_f - y_{target})^2\right] / target_{ratio}{}^2$

2.

$$cost_{final} = \left\{0.04 * LSE\left(route\right) + 0.002 * \left(v - v_{target}\right)^2 + 0.0005 * \left[steering^2 + throttle^2 + brake^2 + throttle * brake\right]\right\} / time\_step\_ratio \quad (1)$$

Phase 03
We connect the Dynamics and costs to Differentiate MPC library to generate an optimal trajectory to our CARLA environment simulator. This process is a bit complicated because the problems with the Pytorch version and the specific way to transcribe the functions are delicate. Our first attempt was to create a Cost function and analyze the car movement.

5 Experiments

Our results were based on a pure-pursuit approach. The first attempt was to simulate the movement in a straight line for a limited time (10 seconds).

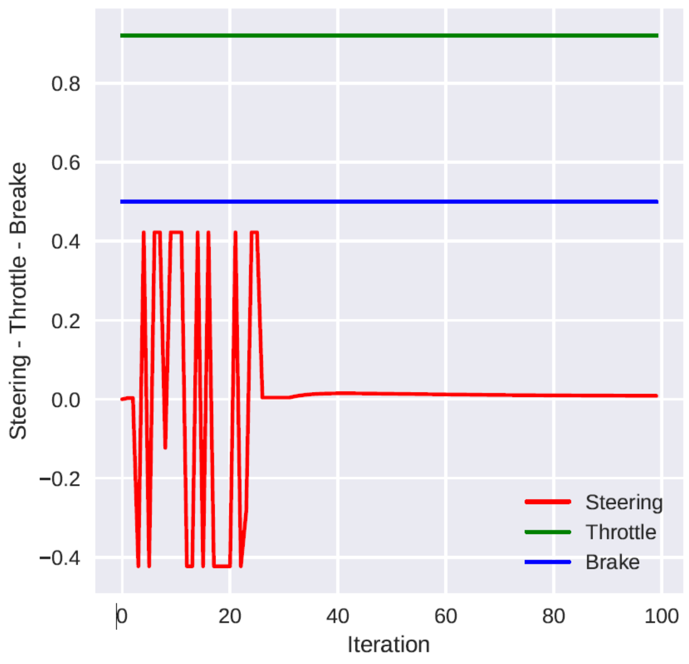

Figure 4: Controls Actuation.

After 25 time steps, approximately the car losing tracking and moves to a random place on the map, the cost function suffers a negative increase, but the control doesn't actuate to bring the car back to the way.
This is a behavior of the car after losing track and deviates from the planning path.

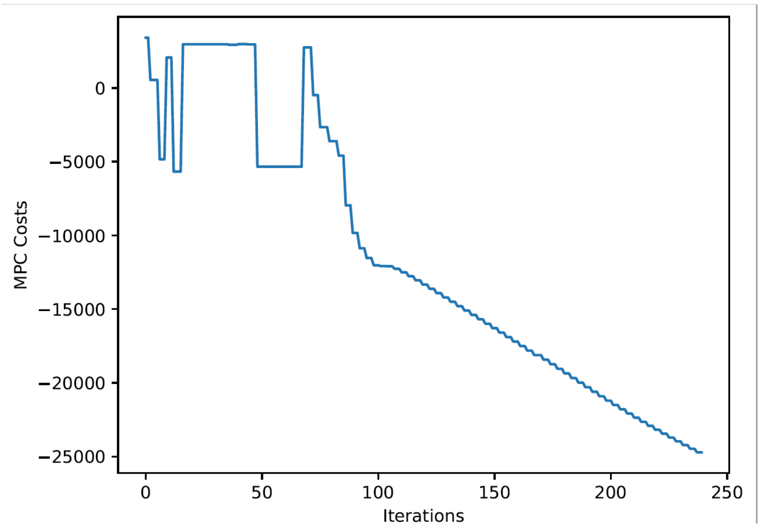

Figure 5: Cost history.

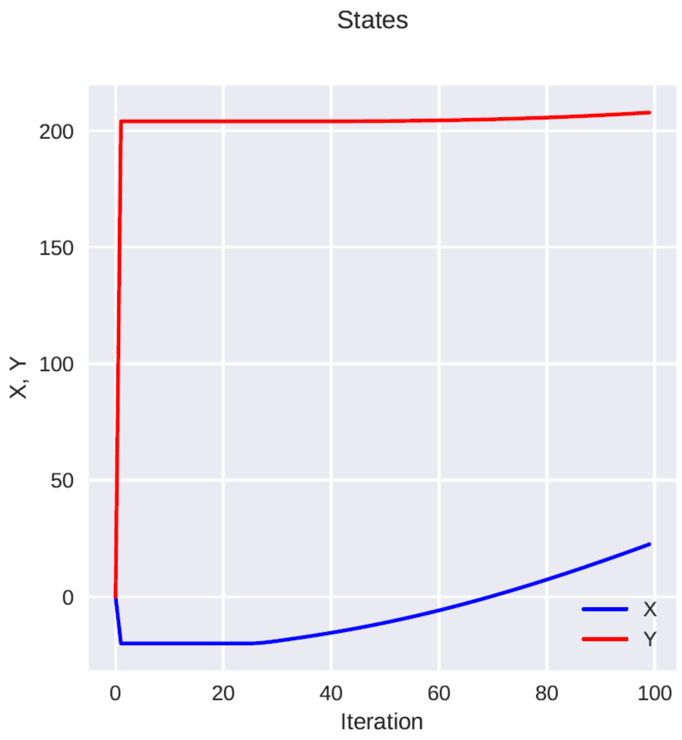

Figure 6: X and Y coordinates.

6 Conclusions and Future Work

Our results didn't reach our expectations, and we are facing problems using the Differentiate MPC library regarding the cost function and insert some reference tracking inside it.
We tried to handle the problem with some tracking error and map the waypoints into the vehicle coordinates. Analyzing figure 6, we concluded that the car follows a straight line until 25 time steps (approximately) and, after losing track and deviates from the reference trajectory at each time step.
The main task to follow in the future is to improve this library to work with more complex dynamical systems, like a bicycle model (the basis for autonomous cars). These systems needed to track some reference trajectory each time, which required the reference required to be included in the cost function.
Adapt the dynamics and cost functions to the library function pattern was a challenge and, unfortunately, doesn't work how we hope. We can find many systems related to MPC, iLQR, and so on to control an autonomous car system, but the control side adapts to the modeled dynamics. In our case, we needed to adjust the dynamics model and cost functions to library patterns.

7 Project video

Link: AA203 Project Video

References

[1] Amos Brandon, Rodriguez Jimenez Ivan Dario, Sacks, Boots, Zico Kolter, Differentiable MPC for End-to-end Planning and Control
[2] YukunXia, Real-time MPC with iLQR for Self-Driving in Carla
[3] Che Kun Law, Darshit Dalal, Stephen Shearrow, Robust Model Predictive Control for Autonomous Vehicle/Self-Driving Cars
[4] Tassa Yuval, Mansard Nicolas, Todorov Emo, Control-Limited Differential Dynamic Programming
[5] Jianyu Chen, Wei Zhan, Masayoshi Tomizuka, Constrained iterative LQR for on-road autonomous driving motion planning
[6] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, Alexey Dosovitskiy, End-to-end Driving via Conditional Imitation Learning