AA203 Final Project Midterm Report

Ricardo Paschoeto[1] and Srikanth Vidapanakal[2]

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

May 7, 2021

1 Motivation

We chose this project by giving the chance to mix the interest of the group members in the subject related to the artificial intelligence area and control theory. The main objective was to look for a project that we can apply the concepts and ideas from the areas above in a real-life situation: path optimization and improvement performance of the autonomous system. And join the knowledge and experience of group members acquired and improved in related courses and training in related fields. Another point to highlight is associated with the technical advancements and opportunities in autonomous systems, Robotics, and Control.

2 Contributions

Below we enumerate our project contributions:

• Our project has based on the paper [1], and it is an opportunity to extend the Theory presented by testing it with another dynamic model system (Actually works on Pendulum and Cartpole dynamics)
• Delivering new simulation data and reinforce concepts over recent techniques such as Differentiable MPC controller.
• The project can be used by researchers/students, in the future, in real-life projects, or basis for depth research;

3 Related work

In the paper [1], the authors present the basis for using the Differentiable Model Predictive Control (MPC) as a policy for reinforcement learning in continuous pair action/state. The objective is to combine the behaviors of model-free and model-based technics. The approach is based on the fact that model-free reinforcement learning suffers from poor sample complexity and generalization, and these could be improved using Differentiable MPC as "expert" to generate qualified data samples to the model-free system. Below is shown the MPC optimization problem:

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

$$x_{t+1} = f\left(x_t, u_t\right), x_1 = x_{init}$$

The authors consider the MPC as a generic policy class $u = \pi\left(x_{init}, C, f\right)$, where C is a cost function, and f is the model dynamics. By differentiating, we can learn the costs and dynamics model in an online policy process. The process to differentiate done by an efficient method provided by the authors for analytically differentiate through an iterative non-convex optimization, computing by an iterative LQR solver. In the end, the cost, and dynamics from an MPC expert have a loss based only on the policy (actions).

Resuming, author's propose is provide a process where the cost and dynamics components can be extracted and analyze on their own (form model-based model MPC approach). The dynamics model and cost now can be learned entirely (by the model-free approach). This process will generate policies more data-efficient than a generic neural network, where the data comes from the "True" model.

4 Project setting – Differential MPC applied to Autonomous Driving

The statements of project settings follow the steps below:

• First, we need to decide the dynamics for our model based Differentiable MPC. For the autonomous system, we have some options: Drone, autonomous car, autonomous robot, and so on. The choices come with a dynamic set with evolves different levels of complexity. We chose Autonomous vehicle driving setup with Carla-based simulation.

• Key idea of our work is to apply differentiable MPC where the cost and dynamics of the Autonomous Vehicle are learnt by optimizing the imitation learning loss using only observed controls as shown below:

$$L = \ E_x[\left\| u_{1:T}\left(x;\theta\right) - u_{1:T}\left(x;\widehat{\theta}\right)\right\|_2^2]$$

Typically the expected dynamics of an autonomous vehicle are described below:

$$x_{(t+1)} = x_t + v_t \cos\left(\varphi_t\right) \ * dt,$$

$$y_{t+1} = y_t + v_t \sin\left(\varphi_t\right) \ * dt,$$

$$\varphi_{t+1} = \varphi_t + \frac{v_t}{L_f}\delta * dt,$$

$$v_{t+1} = v_t + a_t * dt,$$

$$cte_{t+1} = f\left(x_1\right) - y_t + \left(v_t \sin\left(e\varphi_t\right)\ dt\right),$$

$$e\varphi_{t+1} = \varphi_t - \varphi des_t + \left(\frac{v_t}{L_f}\delta_t dt\right).$$

Where:

- x, y – Position
- psi – Orientation
- v – velocity
- CTE – cross track error
- a – Acceleration
- delta – steering angle (range: -25 degrees to +25 degrees)

Goal of this project is that without explicitly defining the dynamics, differentiable MPC controller would learn the dynamics and cost from the expert agent by explicitly imitating the control actions and overall minimizing the imitation loss. Next section describes the details of the Simulation set up and the expert agent.

5 Carla simulator and Conditional Imitation Learning (COIL) setup

Figure 1: CARLA simulator

CARLA is an open-source simulator developed to support, train and validate autonomous vehicle systems. It provides open source, protocols and various scenarios. It also extends support for a Conditional Imitation Learning Agent.

Conditional Imitation Learning Agent

We intend to use conditional imitation learning agent trained on CARLA simulator for our experiments. Below are the measurements and controls provided as part of the dataset:

1. Steer, float
2. Gas, float
3. Brake, float
4. Hand Brake, boolean
5. Reverse Gear, boolean
6. Steer Noise, float
7. Gas Noise, float
8. Brake Noise, float
9. Position X, float
10. Position Y, float
11. Speed, float
12. Collision Other, float
13. Collision Pedestrian, float
14. Collision Car, float
15. Opposite Lane Inter, float
16. Sidewalk Intersect, float
17. Acceleration X,float
18. Acceleration Y, float
19. Acceleration Z, float
20. Platform time, float
21. Game Time, float
22. Orientation X, float
23. Orientation Y, float
24. Orientation Z, float
25. High level command, int (2 Follow lane, 3 Left, 4 Right, 5 Straight)
26. Noise, Boolean (If the noise, perturbation, is activated, (Not Used))
27. Camera (Which camera was used)
28. Angle (The yaw angle for this camera)

Reference: https://github.com/carla-simulator/imitation-learning

The key idea of conditional imitation learning is that an end-to-end learning agent cannot handle the intersection unless explicitly guided by navigational commands.

Reference: http://vladlen.info/papers/conditional-imitation.pdf

6 Preliminary Results

Our first simulations were done to familiarize with the coding process to generate the learning data from Differentiable MPC. We use the sample data from cart-pole to generate data that we can use as a benchmark for our data in the future.

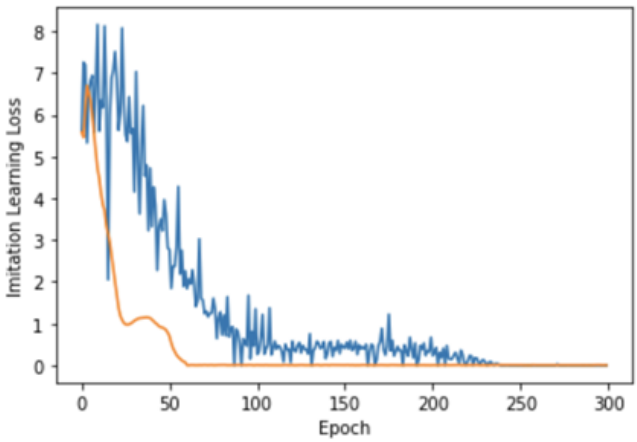

Figure 2: Imitation Learning loss

References

[1] Brandon Amos, Ivan Dario Jimenez Rodriguez, Jacob Sacks, Byron Boots, J. Zico Kolter - Differentiable MPC for End-to-end Planning and Control, Oct 2019

[2] Urban driving with Conditional Imitation Learning: https://arxiv.org/pdf/1912.00177.pdf

[3] Differentiable MPC for End-to-End Planning and Control: https://arxiv.org/pdf/1810.13400.pdf

[4] Neural Network iLQR - https://arxiv.org/pdf/2011.10737.pdf

[5] Model predictive control – sample implementation: https://github.com/sreakdgeek/Model-Predictive-Cor

[6] Carla simulator – Conditional Imitation Learning: https://github.com/carla-simulator/imitation-learning