Performance of MPC vs RL Based Methods for Car Racing

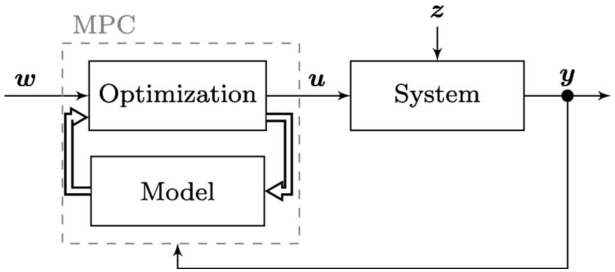
Group 9

Eddie Shi, Fahim Yousuf Choudhury, Tarun S. Ganamur,
Yashvardhan Saravan Nalina

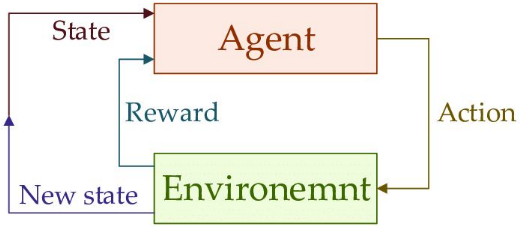
4/23/24

Research Question

- Compare the performance of a MPC based architecture to a model based Reinforcement Learning based architecture for car racing
    - Scenario: Car racing around benchmark track
    - Overall objective: Lap times around benchmark track
        - Secondary objective: Adaptability



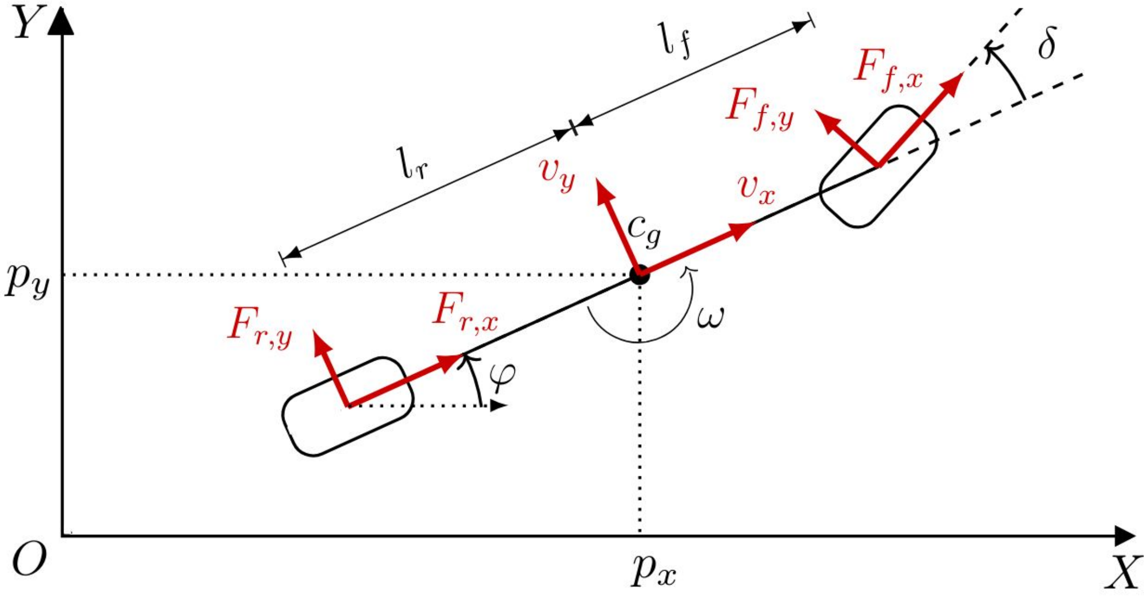
https://www.researchgate.net/figure/Simplified-block-diagram-of-a-MPC-based-control-loop_fig2_353839542



https://www.researchgate.net/figure/Block-diagram-of-reinforcement-learning-RL-control_fig2_344992071

MPC Formulation (Eddie and Fahim)

- Dynamic Bicycle Dynamics:



MPC Formulation (Eddie and Fahim)

- Dynamic Bicycle Dynamics:

$$u = \begin{bmatrix} u[1] \\ u[2] \end{bmatrix} = \begin{bmatrix} \dot{\delta} \\ \dot{v_x} \end{bmatrix}$$

Steering angle velocity
Longitudinal acceleration

$$x = \begin{bmatrix} x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \\ x[6] \\ x[7] \end{bmatrix} = \begin{bmatrix} p_x \\ p_y \\ \delta \\ v_x \\ \omega \\ \dot{\omega} \\ \omega - \delta \end{bmatrix}$$

X Position
Y Position
Steering angle of front wheels
Velocity in x direction
Yaw angle
Yaw rate
Slip angle at vehicle center

MPC Formulation (Eddie and Fahim)

- Dynamic Bicycle Dynamics:

Forward Euler Discretization:

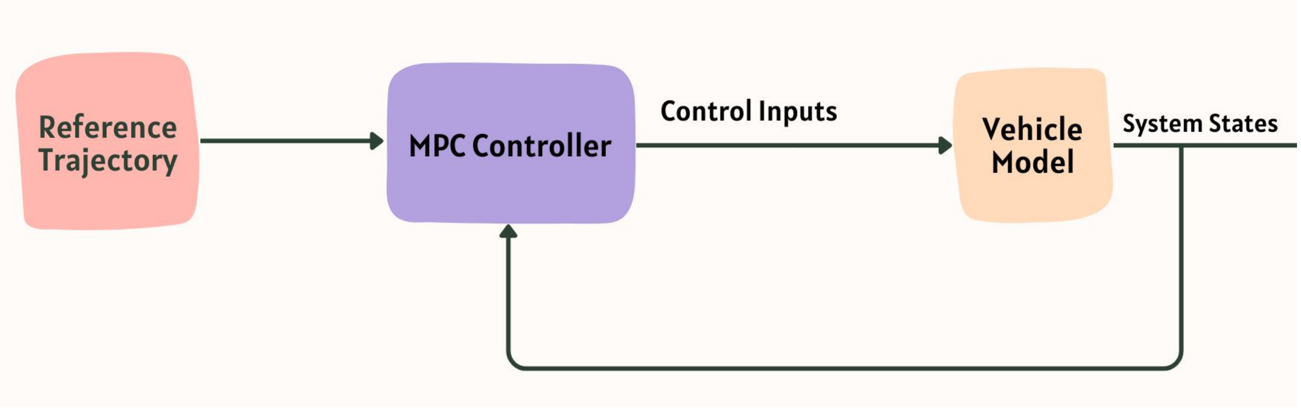
$$x_{k+1} = x_k + \Delta t \cdot \dot{x}_k$$

$$\dot{x} = \begin{bmatrix} \dot{x}[1] \\ \dot{x}[2] \\ \dot{x}[3] \\ \dot{x}[4] \\ \dot{x}[5] \\ \dot{x}[6] \\ \dot{x}[7] \end{bmatrix} = \begin{bmatrix} x[3] \cdot \cos(x[6] + x[4]) \\ x[3] \cdot \sin(x[6] + x[4]) \\ u[0] \\ u[1] \\ x[5] \\ -\frac{\mu \cdot m}{x[3] \cdot I \cdot (l_r + l_f)} \left(l_f^2 \cdot C_{Sf} \cdot (g \cdot l_r - u[1] \cdot h) + l_r^2 \cdot C_{Sr} \cdot (g \cdot l_f + u[1] \cdot h) \right) \cdot x[5] \\ + \frac{\mu \cdot m}{I \cdot (l_r + l_f)} \left(l_r \cdot C_{Sr} \cdot (g \cdot l_f + u[1] \cdot h) - l_f \cdot C_{Sf} \cdot (g \cdot l_r - u[1] \cdot h) \right) \cdot x[6] \\ + \frac{\mu \cdot m}{I \cdot (l_r + l_f)} \cdot l_f \cdot C_{Sf} \cdot (g \cdot l_r - u[1] \cdot h) \cdot x[2] \\ \left(\frac{\mu}{x[3]^2 \cdot (l_r + l_f)} \left(C_{Sr} \cdot (g \cdot l_f + u[1] \cdot h) \cdot l_r - C_{Sf} \cdot (g \cdot l_r - u[1] \cdot h) \cdot l_f \right) - 1 \right) \cdot x[5] \\ - \frac{\mu}{x[3] \cdot (l_r + l_f)} \left(C_{Sr} \cdot (g \cdot l_f + u[1] \cdot h) + C_{Sf} \cdot (g \cdot l_r - u[1] \cdot h) \right) \cdot x[6] \\ + \frac{\mu}{x[3] \cdot (l_r + l_f)} \cdot C_{Sf} \cdot (g \cdot l_r - u[1] \cdot h) \cdot x[2] \end{bmatrix}$$

$\boxed{\dot{x}[6]}$

$\boxed{\dot{x}[7]}$

MPC Controller: (Eddie and Fahim)



MPC Formulation (Eddie and Fahim)

- Cost Function:

$$\underset{\mathbf{u}}{\text{minimize}} \quad \|\mathbf{p}_N - \mathbf{p}^{\mathrm{d}}\|^2_{\mathbf{Q}_1} + \sum^{N-1} \|\mathbf{u}_k - \mathbf{u}_{k-1}\|^2_{\mathbf{Q}_2}$$

- Constraints:

  - ➔ Initial Conditions
  - ➔ Dynamics ($\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k), \ k = 0, \ldots, N-1,$)
  - ➔ Track Limits ($\|\mathbf{p}_k - \mathbf{p}'_k\|^2 \leq (R_g - R_c)^2, \ k = 0, \ldots, N-1,$)
  - ➔ State and Input Constraints($\underline{\boldsymbol{\mu}} \leq \mathbf{u}_k \leq \bar{\boldsymbol{\mu}}, \ k = 0, \ldots, N-1,$ $\underline{\gamma} \leq \mathbf{x}_k \leq \bar{\gamma}, \ k = 0, \ldots, N,$)

MPC Formulation (Eddie and Fahim)

- Parameter Values being used:

Table 1: Parameter Values

Parameter	Value	Parameter	Value
Ts (Sampling Time)	0.033s	N (MPC Horizon)	50
Q_1 (Weight for states)	diag(10,10)	Q_2 (Weight for inputs)	diag(10,10)
R_c (Width of car)	0.24m	R_g (Width of track)	2m
u_l	$(0, -\frac{\pi}{6})$	u_u	$(0, \frac{\pi}{6})$

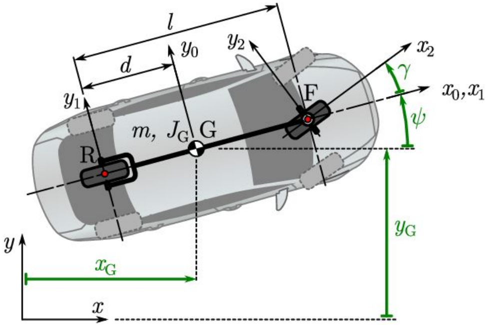
Gym Environment (Tarun and Yash)

- A gym is a standard set of API calls that can be made to communicate between RL agents and the environment
- F1Tenth Gym is an environment used RL and control of racing AVs
  - Original made for scale RC cars to test self-driving algorithms and control
  - Has many environments (racetracks) to test RL algorithms on
  - Built in dynamics model
    - Single Track Bicycle Model
      - Same as MPC architecture

TENTH

f1tenth.org

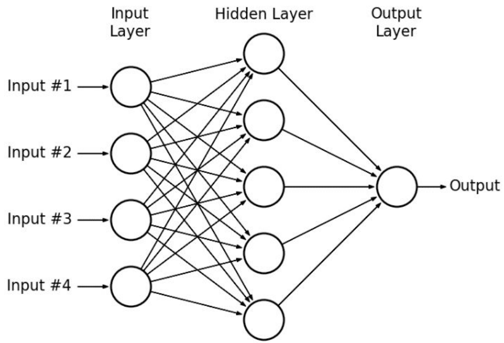
https://www.google.com/imgres?q=f1tenth&imgurl=https%3A%2F%2Ff1tenth.readthedocs.io%2Fen%2Ffoxy_test%2F_images%2Ff1tenth_NX.png&imgrefurl=https%3A%2F%2Ff1tenth.org%2Fbuild.html&docid=Ppu_4KOgDBkvwM&tbnid=FEgC8UzTZqk9wM&vet=12ahUKEwjY0O30vNeFAxVpLUQIHSZfC80QM3oECBQQAA..i&w=1660&h=1589&hcb=2&ved=2ahUKEwjY0O30vNeFAxVpLUQIHSZfC80QM3oECBQQAA



https://www.researchgate.net/figure/Single-track-also-called-bicycle-model-of-an-automobile-with-geometry-and-coordinate_fig10_353730521

Reinforcement Learning Simulation (Tarun and Yash)

- Proximity Policy Optimization (PPO) is an algorithm developed by OpenAi for use in reinforcement learning based architectures
  - Allows to train agents in complex environments while preserving costs in tuning time, training time, and overall performance
- Stable_baselines3 library used to implement PPO for PyTorch
  - Multilayer perceptron architecture



https://www.researchgate.net/figure/A-hypothetical-example-of-Multilayer-Perceptron-Network_fig4_303875065

OpenAi PPO in action

https://openai.com/research/openai-baselines-ppo

Reinforcement Learning Policy Formulation (Tarun and Yash)

- Reward Function:

$$\text{Reward} = V + P + C - S$$

Where,

$$V = \text{vel\_magnitude}$$

$$P = \frac{\text{count}}{\text{total\_waypoints}} \times 0.5$$

$$C = \begin{cases} 0 & \text{if no collision} \\ -1 & \text{if collision occurs} \end{cases}$$

$$S = \begin{cases} 0 & \text{if angle within threshold} \\ 1 & \text{if angle exceeds threshold} \end{cases}$$

- **V:** Represents the reward based on the velocity magnitude of the car.
- **P:** Reflects the reward based on progress towards the next waypoint, encouraging the car to advance along the track.
- **C:** Indicates the penalty for collisions, imposing a negative reward when collisions occur.
- **S:** Represents the penalty for the car spinning beyond a certain angle threshold, leading to the establishment of a negative reward and termination of episode

Challenges

- Environment Selection
  - One of our Primary roadblocks was deciding on the appropriate environment to use that could balance running both MPC and RL frameworks. Our initial options consisted of, MuJoCo, F1Tenth and CARLA.
- Resource Availability
  - Resource availability posed a significant obstacle, as not all team members had access to GPUs, resulting in unexpectedly prolonged computation times.
- Understanding the reinforcement learning architecture as it was novel to everyone in the group
  - Specifically understanding the proximal policy optimization
- MPC formulation challenges
  - Optimization solution can take long time to solve
  - Setting up constraints to fit scenario

Next Steps

- For the RL Agent we aim to further tune our parameters in order to optimize our results; within a predefined training steps, 150,000 steps. This effort is geared towards enhancing the performance and efficiency of our agent's learning process.

- For MPC, we plan to look though the constraints to ensure optimization problem is feasible
    - This will include restructuring constraints and which constraints are hard constraints vs. soft constraints

Thank You!

Questions?