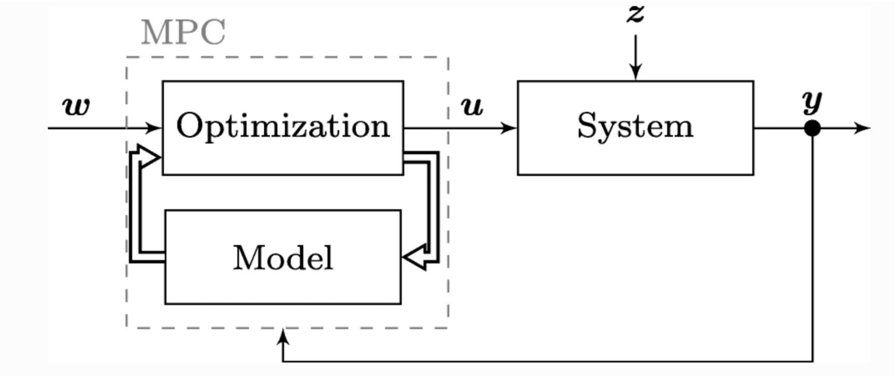
Model Predictive Controls

What is Model Predictive Control ?



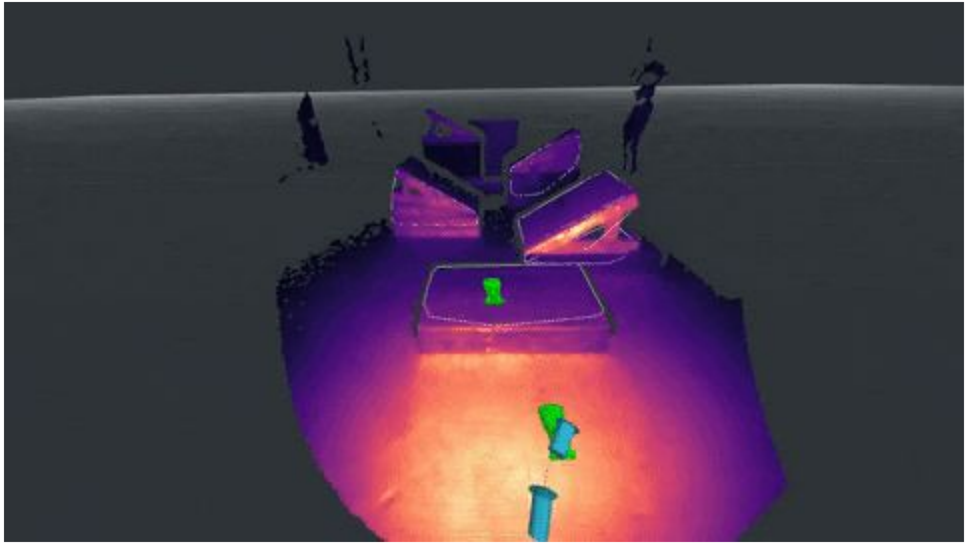
- In MPC we use a simplified model of a system to predict it’s control signals for H timesteps into the future.
- The control signal is obtained by minimizing an objective function while satisfying a set of constraints.
- Only the first control is used after which we optimize again for the next H timesteps.

Examples

Autonomous Driving

- https://github.com/commaai/openpilot/tree/master/selfdrive/controls/lib/lateral_mpc_lib
- https://github.com/commaai/openpilot/tree/master/selfdrive/controls/lib/longitudinal_mpc_lib

Boston Dynamics: ATLAS



"A first person view showing both the perception and planned path. The blue arrows correspond to MPC predictions of the robot's center of mass and momentum as it moves through the course."

-

https://www.bostondynamics.com/resources/blog/flipping-script-atlas

CineMPC


Artistic composition. Fifth sequence.
Instructions
CineMPC
CineMPC-Third Person View
Without Intrinsics

Cost function

We formulate the path optimization problem with the variables as linear velocities v_{t_i} and angualar velocities ω_{t_i} over the time interval $[t_i, t_{i+n}]$.

The cost funtion is written as:

$$\underset{\mathbf{v},\mathbf{w}}{\text{minimize}}(x_N(\mathbf{v},\mathbf{w}) - x_g)^2 + (y_N(\mathbf{v},\mathbf{w}) - y_g)^2 + (\theta_N(\mathbf{v},\mathbf{w}) - \theta_g)^2 \tag{1}$$

Here x_N is a function that takes the vector of velocities $\mathbf{v}$ and angular velocities $\mathbf{w}$ as input and uses the unicycle kinematics model of the vehicle to give the x coordinate of the vehicle at the N-th timestep.

The function y_N and θ_N gives the y and θ of the vehicle in the N-th timestep in the same manner.

This cost function ensures that the N-th position of the vehicle is the closest to the goal configuration of the vehicle.

Bound constraints

Since the autonomous vehicles have a maximum and minimum velocity, angular velocity and acceleration and angular acceleration bounds, we add them as bound constraints to the cost function:

$$v_{min} \leq \mathbf{v} \leq v_{max} \tag{1}$$

$$\omega_{min} \leq \boldsymbol{\omega} \leq \omega_{max} \tag{2}$$

$$a_{min} \leq \mathbf{a} \leq a_{max} \tag{3}$$

$$\alpha_{min} \leq \boldsymbol{\alpha} \leq \alpha_{max} \tag{4}$$

Obstacle avoidance constraints

The obstacle avoidance constraints are given as:

$$d(A_{pos_t}, O^i_{pos_t}) \geq r_a + r^i_o \tag{1}$$

Here A_{pos_t} is the position of the agent at timestep t such that $1 \leq t \leq T$ and $O^i_{pos_t}$ is position the obstacle i at timestep t such that $1 \leq i \leq N$ and $1 \leq t \leq T$. $d(A_{pos_t}, O^i_{pos_t})$ is the euclidean distance between the agent and obstacle i at timestep t. r_a is the radius of the agent and r_o is the radius of the obstacle.

Straight lane boundary constraints

The straight lane constraints are given as:

$$x_{left} \leq s(\mathbf{v}, \boldsymbol{\omega}) \leq x_{right} \tag{1}$$

Here $s(\mathbf{v}, \boldsymbol{\omega})$ is a function that takes the $\mathbf{v}$ and $\boldsymbol{\omega}$ as input and outputs the x and y coordinates of the agent. x_{right} is x coordinate of the right lane boundary and x_{left} is the x coordinate of the left lane boundary since our straight lane is vertical.