Optimization-based Collision Avoidance

Homework 2

Debajyoti Chakrabarti, Ayush Agarwal and Radha Lahoti
Team #7

1 Introduction

Trajectory planning in obstacle-rich environments is a fundamental challenge for autonomous systems, including robots, self-driving cars, and aerial vehicles. A key difficulty lies in modeling collision avoidance constraints, which are typically nonconvex and nondifferentiable, making them hard to handle in real-time optimization frameworks.

Existing optimization-based methods often suffer from major limitations:

1. Use of linear approximations for collision constraints, resulting in uncertainty of error bounds.
2. Modeling of controlled objects as point masses, neglecting their true geometry (this conservative approach can lead to an infeasible solution).
3. Dependence on integer variables to model polyhedral obstacles, which limits its real-time applicability for nonlinear systems.

To address these issues, Zhang et al.[ZLB21] propose a novel method for reformulating collision avoidance constraints into smooth, differentiable forms using strong duality from Convex Optimization. This allows:

1. Exact and smooth reformulations applicable to convex obstacle shapes.
2. Support for both point-mass and full-dimensional controlled objects.
3. Signed distance formulation for minimum-penetration trajectories when collisions are unavoidable.
4. Compatibility with general-purpose nonlinear solvers (e.g., IPOPT).

The proposed method is demonstrated on an autonomous parking task in tight environments. Numerical results show that the approach enables robust, real-time trajectory planning while respecting dynamic constraints.

2 Objective

In this homework, we have implemented both a one-step optimal control and MPC to obtain the optimal collision-free trajectory for reverse parking maneuver. The specific objectives are:

- **Understand and implement** the smooth and exact reformulations of collision avoidance constraints using convex optimization duality.
- **Implement one-shot Optimal Control Problem (OCP) and Model Predictive Control (MPC)** with these constraints alongside dynamics, terminal constraints and state and control input bounds.
- **Demonstrate the autonomous vehicle reverse parking application** presented in the paper [ZLB21].
- **Evaluate performance** in terms of success in collision-avoided parking.

3 Methodology

3.1 Optimization Problem

In [ZLB21], the optimization problem for a full-dimesional object model for a collision free trajectory is given as,

$$\min_{\mathbf{x},\mathbf{u}} \quad J(x_k, u_k) \tag{1}$$

$$\text{s.t.} \quad x_0 = x(0), \quad x_{N+1} = x_F, \tag{2}$$

$$x_{k+1} = f(x_k, u_k), \tag{3}$$

$$h(x_k, u_k) \leq 0, \tag{4}$$

$$-g^\top \mu_k^{(m)} + \left(A^{(m)} t(x_k) - b^{(m)}\right)^\top \lambda_k^{(m)} > d_{min}, \tag{5}$$

$$G^\top \mu_k^{(m)} + R(x_k)^\top A^{(m)\top} \lambda_k^{(m)} = 0, \tag{6}$$

$$\left\| A^{(m)\top} \lambda_k^{(m)} \right\|_* \leq 1, \tag{7}$$

$$\lambda_k^{(m)} \succeq_{\mathcal{K}^*} 0, \quad \mu_k^{(m)} \succeq_{\bar{\mathcal{K}}^*} 0, \tag{8}$$

$$\text{for } k = 0, \ldots, N, \quad m = 1, \ldots, M.$$

Here, (1) is the cost, (2) are the initial and terminal constraints respectively, (3) represents the dynamics constraints, (4) represents the bounds on the states and control inputs, N corresponds to the length of the time horizon. The equations (5), (6), (7), (8) correspond to the obstacle avoidance constraints where $\lambda_k^{(m)}, \mu_k^{(m)}$ are the dual variables associated with the collision avoidance constraints and M is the number of obstacles in the path. Note that in the case of the point mass approximation for modeling the object (car), **all** $\mu_k^{(m)}$ **would be** 0.

3.2 Dynamics

The car is described by the classical kinematic bicycle model (1), which is well-suited for velocities used in typical parking scenarios. The state vector $x = [X, Y, \theta, v]^T$. The states (X, Y) correspond to the center of the rear axis, while θ is the yaw angle with respect to the x-axis and v is the velocity with respect to the rear axis. The control inputs $u = [a, \delta]^T$ consist of the steering angle δ and the acceleration a.

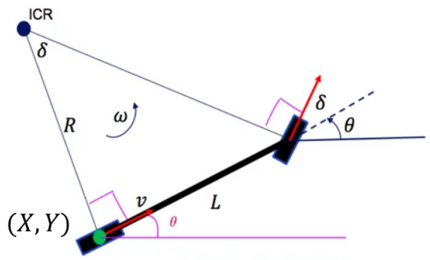

Figure 1: Bicycle Model

The continuous time system dynamics is given by,

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\theta} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \frac{v\tan(\delta)}{L} \\ a \end{bmatrix} \tag{9}$$

where $L = 2\,\text{m}$ is the wheel base of the car. Discretization is done via forward Euler integration as follows,

$$\begin{bmatrix} X_{k+1} \\ Y_{k+1} \\ \theta_{k+1} \\ v_{k+1} \end{bmatrix} = f(x_k, u_k) = \begin{bmatrix} X_k + v_k \cos(\theta_k)\Delta t \\ Y_k + v_k \sin(\theta_k)\Delta t \\ \theta_k + \frac{v_k \tan(\delta_k)}{L}\Delta t \\ v_k + a_k \Delta t \end{bmatrix} \tag{10}$$

Vehicle and Obstacle Representation: The vehicle is represented as a rotated and translated rectangle using a convex set transformation:

$$E(x_k) = R(x_k)B + t(x_k),$$

where B is the base shape of the vehicle. For this project, the set is defined as a rectangle of dimension $L \times \frac{L}{4}$.
Obstacles are modeled as rectangles. Two closed-packed rectangles of dimension 8 m $\times$ 3 m are considered for the reverse parking problem. They are centered at locations $(4, -4)$ m and $(14, -4)$ m respectively.

3.3 Obstacle avoidance constraints

The obstacle avoidance constraints are explained below. Table 1 summarizes the role of each constraint.

1. **Constraint** (5)**: Separation condition**

$$-g^\top \mu_k^{(m)} + \left(A^{(m)} t(x_k) - b^{(m)}\right)^\top \lambda_k^{(m)} > d_{\min}$$

Purpose: Ensures a minimum distance $d_{\min}$ between the object and the m-th obstacle at time step k.

Here $A^{(m)}x \leq b^{(m)}$: halfspace representation of the obstacle; $t(x_k)$: position of the object at time k; $\lambda_k^{(m)}$, $\mu_k^{(m)}$: dual variables for the obstacle and object, respectively; g: object geometry vector, which in our case is $g = [L, L/4, L, L/4]^T$.

This constraint ensures that a valid dual certificate of separation exists, meaning the robot is outside the obstacle by at least $d_{\min}$. Note that in the case of the point mass approximation for modeling the car, **all** $\mu_k^{(m)}$ **would be** 0.

2. **Constraint** (6)**: Dual Equilibrium**

$$G^\top \mu_k^{(m)} + R(x_k)^\top A^{(m)\top} \lambda_k^{(m)} = 0$$

Purpose: Enforces alignment between the robot's shape normals and obstacle contact normals via dual variables.

Here $R(x_k)$: rotation matrix of the object body, which in 2D is,

$$R(x_k) = \begin{bmatrix} \cos\theta_k & -\sin\theta_k \\ \sin\theta_k & \cos\theta_k \end{bmatrix};$$

G: face normal matrix of the object geometry, which in our case is,

$$G = \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

This constraint ensures force balance in dual space: the object's contact configuration must align with that of the obstacle.

Constraint	Role	Interpretation
(5)	Separation	Enforces robot is at least $d_{\min}$ away from obstacle
(6)	Dual equilibrium	Aligns object and obstacle normals
(7)	Norm bound	Bounded and normalized dual variables
(8)	Dual feasibility	Ensures duals are non-negative

Table 1: Role of the obstacle avoidance constraints

3. **Constraint ((7): Dual Norm Bound**

$$\left\| A^{(m)\top} \lambda_k^{(m)} \right\|_* \leq 1$$

Purpose: Constrains the dual variable magnitude to remain within the unit ball of the dual norm $\|\cdot\|_*$, which depends on the obstacle's shape. This prevents unbounded dual values and ensures dual feasibility relative to the obstacle geometry.

4. **Constraint** (8)**: Dual Nonnegativity**

$$\lambda_k^{(m)} \succeq_* 0, \quad \mu_k^{(m)} \succeq_{\bar{*}} 0$$

Purpose: Enforces non-negativity of dual variables, as required in the dual formulation of inequality constraints.

In our case, $\lambda_k^{(m)} \geq 0$: for obstacle halfspaces; $\mu_k^{(m)} \geq 0$: for object geometry support functions.

These conditions guarantee that only active constraints contribute to the dual optimization.

3.4 Cost

Our control objective is to navigate the vehicle to the goal position in a **reverse parking maneuver** while avoiding excessive control inputs. We combine these competing goals into our cost function as a weighted sum of the form

$$J(x_k, u_k) = (x_N - x_{goal})^T Q_f (x_N - x_{goal}) + \sum_{k=0}^{N-1} (x_k - x_{goal})^T Q (x_k - x_{goal}) + u_k^T R u_k \tag{11}$$

where, $Q_f = Q_f^T \succeq 0$, $Q = Q^T \succeq 0$, and $R = R^T \succeq 0$ are weighting factors.
In our work, the choice of the weight matrices are as follows,
$Q = \text{diag}\{[22, 22, 10, 20]\}$, $Q_f = \text{diag}\{[40, 40, 10, 20]\}$ and $R = \text{diag}\{[0.1, 0.1]\}$.

3.5 Initial and Terminal Constraints and Bounds

The initial state vector of the object is a zero vector. The vehicle needs to reach the target coordinate of $(9, -4)$ m, with orientation, $\theta = \pi/2$ and velocity, $v = 0$ m/s.
The velocity of the vehicle is bounded by, $v \in [-10, 10]$ m/s. The control input constraints are given by, $\delta \in [-1, 1]$ rad and $a \in [-2, 2]$ m/s. We impose the terminal constraint in the one-shot OCP problem but remove it in the MPC, since imposing it makes it difficult to find a feasible solution for the MPC. The target is achieved in the MPC case by using the terminal cost, which is why we set the Q_f slightly higher than Q.

3.6 Problem Setup

The collision avoidance problem is solved using both a one-shot Optimal Control problem (OCP) and a Model Predictive Control (MPC) setup. The one-shot OCP was solved with a time horizon of $T = 10$ seconds and a discretization step size of $dt = 0.2$ seconds. In case of MPC formulation, the horizon is chosen as $N = 50$ and the corresponding time $T = 10$ seconds (i.e., the step size $dt = 0.2$ seconds). The optimal control problem for the one-step Optimal Control and in each iteration of MPC is solved using IPOPT via the CasaDi framework in Python.

3.7 Choice of Initial Guess

Since the given trajectory design problem is nonlinear, a proper initial guess needs to be provided as *warm starting*. For single-shot OCP, a linear interpolation of the state trajectory (between initial and final states) and a zero control input trajectory are provided to the solver. For the MPC formulation, the initial guess of state and control inputs is chosen as zero vectors at the first iteration. For the subsequent iterations of MPC, the initial guesses are provided as the solution of state and control inputs from the previous iteration.

4 Assumptions

While our implementation is based on the work by Zhang et al. [ZLB21], we make the following simplifying assumptions to tailor the methodology to our project constraints:

- **Euclidean norm and standard inequalities:** We use the Euclidean norm $\|\cdot\|_2$ instead of general cone norms, and replace conic inequalities with element-wise inequalities since we are using polyhedral shapes for object and obstacle.
- **No steering rate constraints:** We do not include constraints on the rate of change of the steering input.
- **Fixed discretization step:** A constant time step is used for discretizing the system dynamics, rather than optimizing the step size as suggested in the paper.
- **Minimum-time objective omitted:** The cost function does not include a term for minimizing the total maneuver time.
- **Existence of a collision-free trajectory:** To simplify the problem, we have modeled our system such that a collision-free trajectory exists, and hence we have not implemented the minimum penetration trajectory case.

5 Results

Figure 2 shows the trajectory obtained for the one-shot OCP using the point-mass model for the car. Figure 3 shows the trajectory obtained for the one-shot OCP using the full rectangle model for the car. We see that the trajectory for the point-mass model goes closer to the obstacles compared to the full rectangle model as expected. Further, since we impose the goal state as a terminal constraint, we see that the car reaches the goal state exactly. Next, the trajectory of the vehicle using MPC for the point mass model and the full rectangle model for the car can be seen in Figure 4 and Figure 5respectively. Note that here the car doesn't reach the goal state exactly since the terminal constraint is removed and the goal is achieved using the terminal cost. In figure 6, we plot the evolution of state variables (X, Y, θ, v), and control variables (a, δ) with time for the full-dimensional vehicle model under MPC.

6 Discussion

In this section, we will list some observations based on the obtained numerical results and how these results differ from those in [ZLB21].

- Both one-shot OCP and MPC successfully drive the vehicle to goal position while obeying constraints on the control inputs.
- In the one-shot optimal control problem (OCP), we can enforce the goal state as a hard terminal constraint and still obtain a feasible solution. However, in the model predictive control (MPC) setting, we had to relax this condition by including the goal only as a terminal cost with a higher weight. Enforcing the terminal constraint exactly in MPC made the problem infeasible for reasonable horizon lengths. This behavior is both interesting and expected. In the one-shot OCP, the optimizer has full

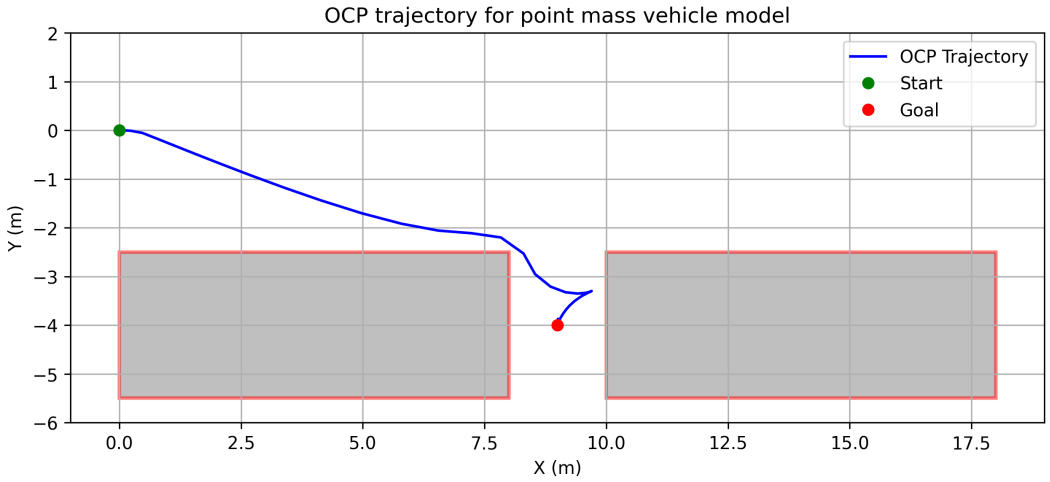

Figure 2: Trajectory of point mass vehicle model with optimization based collision avoidance using OCP

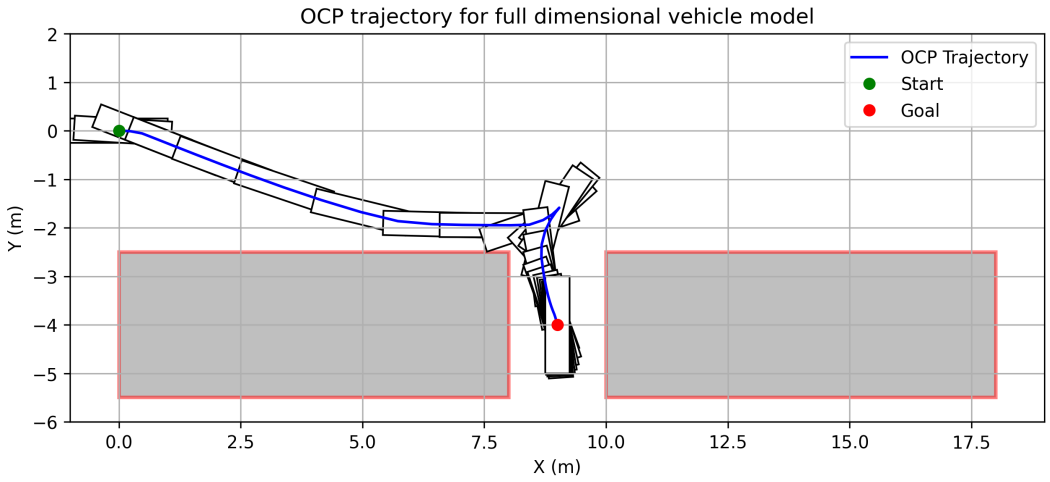

Figure 3: Trajectory of full dimensional vehicle model with optimization based collision avoidance using OCP

knowledge of the trajectory and can distribute control efforts globally across the entire time horizon to meet the terminal condition. In contrast, MPC optimizes over a shorter, receding horizon and re-solves at every step. Imposing a hard terminal constraint at each iteration forces the system to reach the goal state in a fixed number of steps from its current position—regardless of how far it is or what dynamic and obstacle constraints it faces—which can easily lead to infeasibility. Hence, using a soft terminal cost in MPC is a common and practical compromise to guide the system toward the goal without over-constraining the problem.

- Unlike [ZLB21], the objective function in above simulation is a function of the pose error. Hence, MPC initially drives the vehicle towards the goal position, see Fig.4. This may be in contrast to the time minimization problem in [ZLB21].
- The absence of hard constraints on Δa_k, $\Delta\delta_k$ and exclusion of Δu_k from the objective function enables MPC to switch between maximum and minimum of the control inputs. We may add constraints on those or a corresponding cost in order to have smoother trajectories.

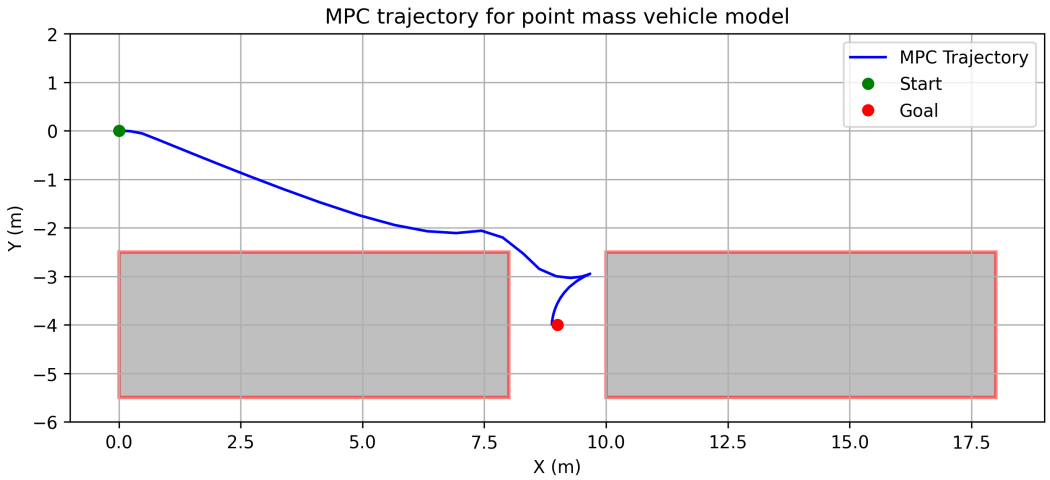

Figure 4: Trajectory of point mass vehicle model with optimization based collision avoidance using MPC

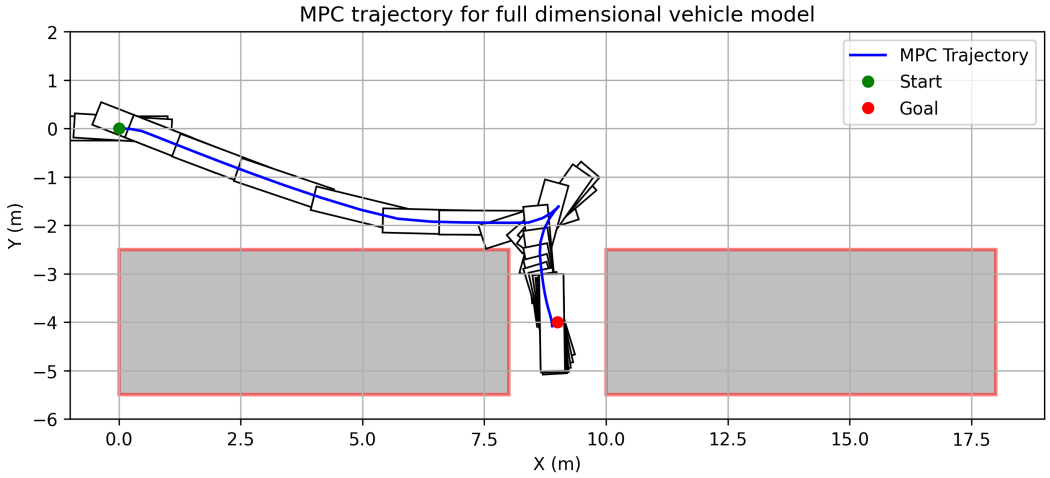

Figure 5: Trajectory of full-dimensional vehicle model with optimization based collision avoidance using MPC

7 Conclusion and Future Work

The collision avoidance framework proposed by Zhang et al. [ZLB21] has been successfully reproduced in this homework. Our results confirm the real-time feasibility of the method for planning parking trajectories in constrained environments. We demonstrated the generation of collision-free trajectories for the autonomous parking problem using both a point-mass model and a full-dimensional model of the vehicle. Specifically, we implemented and solved the reverse parking problem within both a one-step optimal control and a model predictive control (MPC) framework.

For the final report and future development of the project, we plan to extend our work in the following directions:

- **Minimum-penetration trajectory generation:** Formulate and solve the signed-distance-based optimization problem for both point-mass and full-dimensional vehicle models, allowing soft constraint violations when collision-free solutions are infeasible.
- **Dynamic obstacles:** Replace static obstacles with time-varying (moving) obstacles and reformulate the collision avoidance constraints accordingly.

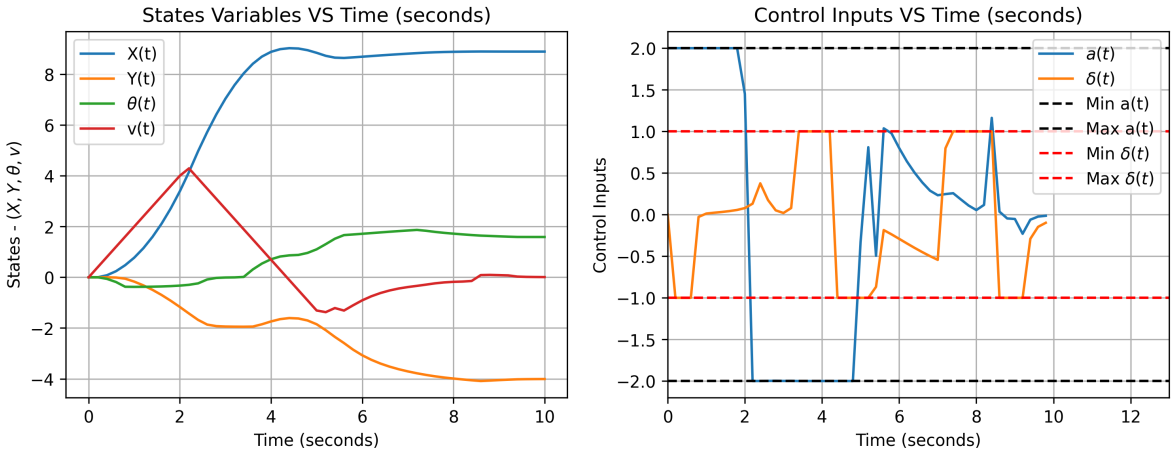

Figure 6: State and control trajectories of the full-dimensional vehicle model under MPC

- (Time permits) **Robustness to disturbances:** Incorporate process noise and model uncertainties into the vehicle dynamics, and formulate a chance-constrained optimal control problem to ensure probabilistic safety.

References

- X. Zhang, A. Liniger, F. Borrelli, "Optimization-Based Collision Avoidance," IEEE TCST, 2021. `https://github.com/XiaojingGeorgeZhang/OBCA`

Code

- One-shot OCP:
  1. Reverse Parking OCP (point mass)
  2. Reverse Parking OCP (full dimensional)
- MPC:
  1. Reverse Parking MPC (point mass)
  2. Reverse Parking MPC (full dimensional)

Media

- GIF for One-shot OCP (full dimensional)
- GIF for MPC (full-dimensional)