Model Predictive Control

Rahul Sajnani
20171056
International Institute of Information Technology, Hyderabad, India
rahul.sajnani@research.iiit.ac.in

1 Introduction

Model Predictive Control deals with determines a trajectory that a robot must follow in order to reach its goal and avoid obstacles. This is formulated as an optimization problem wherein we optimize for the velocities at each time step. Section 2 defines all the assumptions used in this assignment. Section 3 deals with the problem formulation and steps followed to solve the problem. Section 4 displays the results in different scenarios of the Model Predictive Control (MPC) algorithm.

2 Assumptions

For this assignment we make the following assumptions:

1. Environment is **known**: Model Predictive Control (MPC) is a planning problem and assumes that the obstacles positions are known during planning.
2. Obstacles are static and circular whose radius is known.
3. The robot is holonomic with given initial and goal positions.

3 Problem formulation and solution

Let $\hat{\mathbf{X}}$ and $\hat{\mathbf{Y}}$ be the velocity vectors where $\hat{\mathbf{X}}, \hat{\mathbf{Y}} \in \mathbb{R}^n$.

$$\hat{\mathbf{X}} = \begin{bmatrix} \dot{x}_0 & \dot{x}_1 & \dot{x}_2 & \dots & \dot{x}_{n-1} \end{bmatrix}^T \tag{1}$$

n are the total number of time steps to plan. The goal of MPC is to predict the velocity commands at each time step $\hat{\mathbf{X}}, \hat{\mathbf{Y}}$. Let (x_0, y_0) be the initial positions of the robot and (x_g, y_g) be the position of the goal. Given the maximum velocity v_{max}, the predicted controls should ensure that the maximum velocity and obstacle constraints are not violated.

3.1 Without obstacles

We begin with formulating the MPC problem without obstacles. Without obstacles we only have to take care of the velocity constraints and minimize the goal cost $\mathcal{L}$.

$$\mathcal{L}(\hat{\mathbf{X}}, \hat{\mathbf{Y}}) = ((x_0 - x_g) + (\dot{x}_0 + \dot{x}_1 + \dot{x}_2 \dots \dot{x}_{n-1})\Delta t)^2 + ((y_0 - y_g) + (\dot{y}_0 + \dot{y}_1 + \dot{y}_2 \dots \dot{y}_{n-1})\Delta t)^2 \tag{2}$$

Here, $\dot{x}_t$ and Δt are the velocity at time t and the time for which that velocity is applied respectively. Note that the position of the robot after time t is as follows:

$$x_t = x_0 + (\dot{x}_0 + \dot{x}_1 + \dot{x}_2 ... \dot{x}_{t-1})\Delta t \tag{3}$$

We can vectorize the above objective in the following manner:

$$(\dot{x}_0 + \dot{x}_1 + \dot{x}_2 ... \dot{x}_{t-1})\Delta t = \hat{\mathbf{X}}^T \mathbf{1}_t \mathbf{1}_t^T \hat{\mathbf{X}} \Delta t^2 \tag{4}$$

$$\mathbf{1}_t = \begin{bmatrix} 1 & 1 & ... & 0 \end{bmatrix}^T \tag{5}$$

$$\mathbf{A} = \mathbf{1}_t \mathbf{1}_t^T \Delta t^2 \tag{6}$$

Here, $\mathbf{1}_t$ vector has $1s$ filled till time *t* followed by $0s$.

$$2(x_0 - x_g)(\dot{x}_0 + \dot{x}_1 + \dot{x}_2 ... \dot{x}_{t-1})\Delta t = 2(x_0 - x_g)\Delta t \mathbf{1}_t^T \hat{\mathbf{X}} \tag{7}$$

$$q_1 = 2(x_0 - x_g)\Delta t \mathbf{1}_t \tag{8}$$

$$2(x_0 - x_g)(\dot{x}_0 + \dot{x}_1 + \dot{x}_2 ... \dot{x}_{t-1})\Delta t = q_1^T \hat{\mathbf{X}} \tag{9}$$

$$C_1 = (x_0 - x_g)^2 \tag{10}$$

$$\mathcal{L}(\hat{\mathbf{X}}) = (x_0 - x_g)^2 + 2(x_0 - x_g)(\dot{x}_0 + \dot{x}_1 + ... \dot{x}_{n-1})\Delta t + ((\dot{x}_0 + \dot{x}_1 + ... \dot{x}_{n-1})\Delta t)^2 \tag{11}$$

Our vectorized objective becomes as follows:

$$\mathcal{L}(\hat{\mathbf{X}}, \hat{\mathbf{Y}}) = \hat{\mathbf{X}}^T \mathbf{A} \hat{\mathbf{X}} + q_1^T \hat{\mathbf{X}} + C_1 + \hat{\mathbf{Y}}^T \mathbf{B} \hat{\mathbf{Y}} + q_2^T \hat{\mathbf{Y}} + C_2 \tag{12}$$

$$s.t.\ \hat{\mathbf{Y}} \leqslant v_{max} \tag{13}$$

$$-v_{max} \leqslant \hat{\mathbf{Y}} \tag{14}$$

$$\hat{\mathbf{X}} \leqslant v_{max} \tag{15}$$

$$-v_{max} \leqslant \hat{\mathbf{X}} \tag{16}$$

3.2 With obstacles

In the obstacle scenario, we need to add an additional constraint to avoid the obstacles. The following is the obstacle constraint for **every** time *t*:

$$(x_0 - x_{obs} + (\dot{x}_0 + \dot{x}_1 + ... \dot{x}_{t-1})\Delta t)^2 + (y_0 - y_{obs} + (\dot{y}_0 + \dot{y}_1 + ... \dot{y}_{t-1})\Delta t)^2 \geqslant R^2 \tag{17}$$

Here, (x_{obs}, y_{obs}) is the position of the obstacle and R is the radius of the obstacle. Note that the above constraint is not a linear constraint. To linearize the above constraint we use the taylor series expansion around the vectors $\hat{\mathbf{X}}^*, \hat{\mathbf{Y}}^*$ which are computed by performing the optimization initially without the obstacles and adding a random vector containing values belonging in [0, 0.2]. This randomization is added so that the optimizer does not receive the same gradients by following the constraints. This constraint is as follows (displaying just for x part for brevity) ignoring the higher order terms in the taylor series expansion:

$$f(\hat{\mathbf{X}}^*) + \nabla f(\hat{\mathbf{X}}^*)(\hat{\mathbf{X}} - \hat{\mathbf{X}}^*) \tag{18}$$

$$= \hat{\mathbf{X}^*}^T \mathbf{A} \hat{\mathbf{X}}^* + q_1^T \hat{\mathbf{X}}^* + C_1^{obs} + (2\mathbf{A}\hat{\mathbf{X}}^* + q_1)^T (\hat{\mathbf{X}} - \hat{\mathbf{X}}^*) \tag{19}$$

$$f(\hat{\mathbf{X}}^*) + \nabla f(\hat{\mathbf{X}}^*)(\hat{\mathbf{X}} - \hat{\mathbf{X}}^*) + f(\hat{\mathbf{Y}}^*) + \nabla f(\hat{\mathbf{Y}}^*)(\hat{\mathbf{Y}} - \hat{\mathbf{Y}}^*) \geqslant R^2 \tag{20}$$

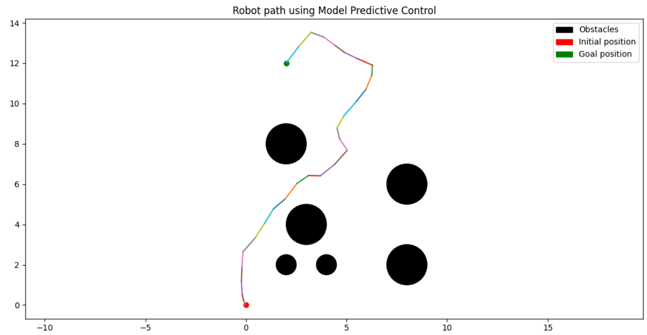
Equation (20) gives the linearized constraint for Quadratic Problem solution. **Note the constraint is applied for every time t.**

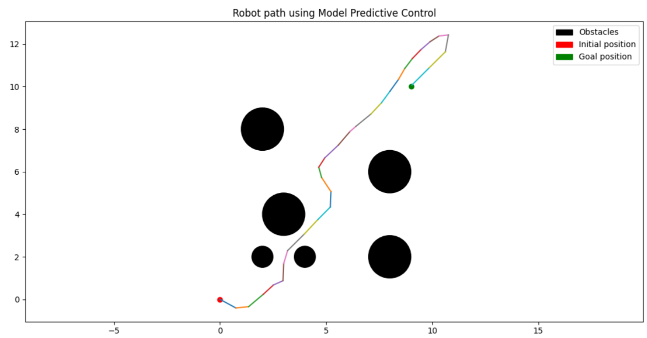
3.3 Trust region

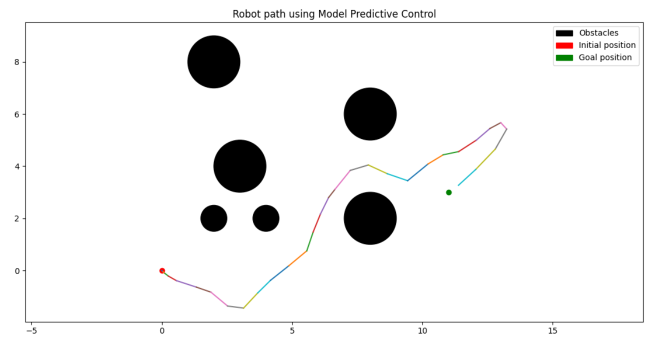
After solving the optimization task we compute the mean absolute error difference between the initialized $\hat{\mathbf{X}}^*$ and optimized $\hat{\mathbf{X}}$. If the error is above a threshold ϵ. Then we reinitialize the $\hat{\mathbf{X}}^*$ and re-optimize to ensure that the solution is within the trust region.

4 Results

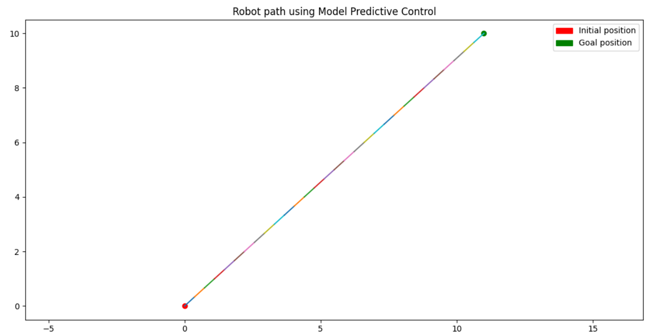
4.1 With obstacles







4.2 Without obstacles



5 Deliverables

The code performs the MPC optimization using the *cvxpy* library. It also allows **a live plotting feature**. All the code is in the *src* directory. Screenshots and gifs are in the *screenshots* directory.

5.1 Contributions

MPC, plotter, and report - Rahul Sajnani