RPN - Assignment #2

Team #3 - Makkhi-2

MPC - Model Predictive Control

Introduction

Model predictive control (**MPC**) is an advanced method of process **control** that is used to **control** a process while satisfying a set of constraints. It has been in use in the process industries in chemical plants and oil refineries since the 1980s.
For our assignment, we used the concept of model predictive control and applied it to a robot navigating in a scene, aiming to reach a certain end location or goal point from a given start point.

Theory

The objective is given by

$$min(x_n - x_g)^2 + (y_n - y_g)^2$$

But is subject to the following constraints

1. **Robot Model constraints:**

$$x_t + dt = x_t + v_{xt}dt$$
$$y_t + dt = y_t + v_{yt}dt$$

2. **Velocity constraints:**

$$v_{min} \leq v_{xi} \leq v_{max} \forall i \in [1;n]$$
$$v_{min} \leq v_{yi} \leq v_{max} \forall i \in [1;n]$$

3. **Obstacle avoidance constraints**

$$(x_n - x_{obs})^2 + (y_n - y_{obs})^2 \geq R^2 \quad \text{where} \quad R = r_{obstacle} + r_{robot}$$

The code was run for 2 cases
(i) With obstacles
(ii) Without obstacles

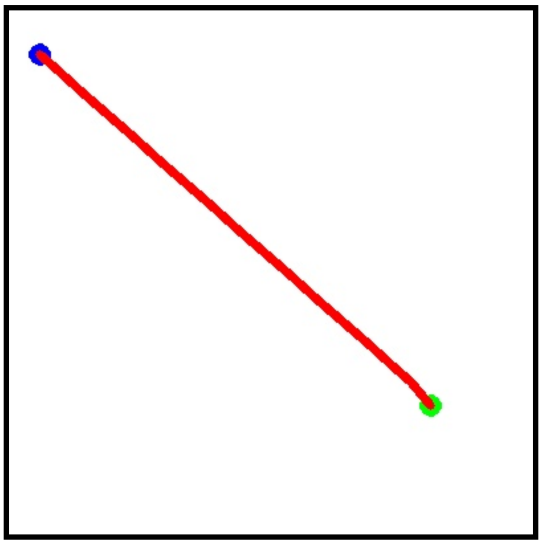
Outline of code

1) Create an empty image. This empty image represents our world. Now, we'll add the start point, end (goal) point, and obstacles to it.

2) Mark the start and end points on it. This will be our world. The start point and end points are defined as a 2-tuple in the main code, and are marked in the image using red and green coloured (small) circles (having a radius of 5) respectively, by using the cv2.circle() function.
3) (If needed), add obstacles to the scene. By using in-built openCV functions like cv2.circle().
4) For the without obstacles case, all we have to do now, is to add the velocity constraints (i.e. v_{min} < x_dot < v_{max}) and the robot equation i.e. x(n+1) = x(n) + x_dot * delta_t
5) Once all the constraints have been appended to the constraints array, the array is passed to the cvxpy.solve() function, which is an in-built python optimization solver for convex optimization problems.
6) For the with obstacles case, some additional effort needs to be put in, as we have to include the obstacle avoidance constraints too. The obstacle avoidance constraints ensure that the robot does not choose a path which will collide with the obstacle.

Results

Without Obstacles - The blue point marks the start & the green point marks the end point. (Link)



With Obstacles - The blue point marks the start and the green point marks the end point. (Link)