PLANNING AND NAVIGATION

Assignment 2

Haripraveen Subramanian, 2018102031
Loay Rashid, 2018102008

Given:

- We consider an omni-wheel robot, hence it is holonomic.
- Let the robot's initial location be (x_0, y_0), and it's goal location be given by (x_g, y_g)
- For a robot at keypoint n, the goal reaching cost is given by:

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

- The robot model is given by:

$$x_{t+dt} = x_t + v_{xt} * dt$$
$$y_{t+dt} = y_t + v_{yt} * dt$$

- The velocity constraints are given by:

$$0 \leq v_{xi} \leq v_{\max} \quad \forall i \in [1, n]$$
$$0 \leq v_{yi} \leq v_{\max} \quad \forall i \in [1, n]$$

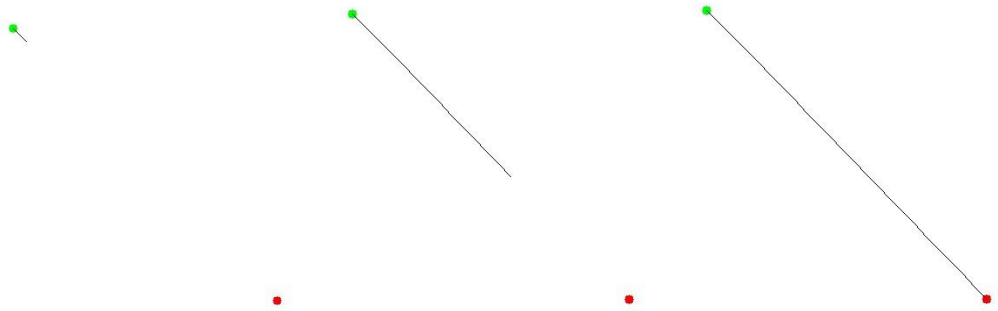
- Several obstacles are there . Obstacles are circular and centre and radius is given

Model Predictive Control:

Model predictive control (MPC) is an advanced method of process control that is used to control a process while satisfying a set of constraints.The general design objective of model predictive control is to compute a tra-jectory of a future manipulated variable to optimize the future behaviour of the plant output . Also MPC has the ability to anticipate future events and can take control actions accordingly.

Without Obstacles:

Without obstacles, the problem becomes a simple case of optimizing the goal cost while maintaining the constraints. We used the python library 'cvxpy' for optimizing.
The constraints that we added was mainly for velocity here maintaining that all velocities should be in the range vmin and Vmax.
Our outputs are attached below. Our submission includes all the results and a video showing the robot's progress. The green point is the start location, the red point is the goal location.
The video showing the robot's progress can be found at:
https://drive.google.com/file/d/1TGPI2lP1Ck9BMYkkpVPQQv4X7ss53d0p/view?usp=sharing



With Obstacles:

We are required to include circular obstacles. This is equivalent to adding the following constraint to our optimization problem:

$$(x_i - x_o)^2 + (y_i - y_o)^2 \geq r_1^2 \quad \forall i \in [1, n]$$

Where r is the radius of the obstacle and (x_o, y_o) is the obstacle center. Multiple constraints, identical to the one above would have to be added for each obstacle.

However, directly adding these constraints makes the optimization problem non-convex.
The objective function can be written as:

$$(x_0 + (\dot{x_0} + \dot{x_1} + \dot{x_2})\delta t - x_g)^2 + (y_0 + (\dot{y_0} + \dot{y_1} + \dot{y_2})\delta t - y_g)^2$$

We linearize this objective using multivariate taylor series expansion to get: (only x-term shown)

$$f(\dot{x_0} + \dot{x_1} + \dot{x_2}) = f(\dot{x_0^*} + \dot{x_1^*} + \dot{x_2^*}) + \begin{bmatrix} \frac{\delta f}{\delta \dot{x_0}} & \frac{\delta f}{\delta \dot{x_1}} & \frac{\delta f}{\delta \dot{x_2}} \end{bmatrix}_{\dot{x_0}, \dot{x_1}, \dot{x_2},} \begin{bmatrix} (\dot{x_0} - \dot{x_0^*}) \\ (\dot{x_1} - \dot{x_1^*}) \\ (\dot{x_2} - \dot{x_2^*}) \end{bmatrix}$$

We take only the first term of taylor series in our implementation.

We calculate $\frac{\delta f}{\delta \dot{x_0}}$ using this formula. We add this constraint for all points till n.

$$\frac{\delta f}{\delta \dot{x_0}} = 2\Delta t((x_0 - x_{ob}) + (\dot{x_0} + \dot{x_1} + \dot{x_2})\Delta t)$$

$$= 2\Delta t((x_0 - x_{ob}) + 2\Delta t^2 \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \dot{x_0^*} \\ \dot{x_1^*} \\ \dot{x_2^*} \end{bmatrix}$$

We then run this approximation many times. Intitializing x* values as zero and updating them on each iteration . we added a termination condition that when loss difference between 2 subsequent iterations is less than 1 we will break. This value was arbitrary.

The video containing the robot's progress can be found at:
https://drive.google.com/file/d/1YKqq4w4sr-CIF-yIn6tD8L30vlFp1mS4/view?usp=sharing