MPC controller for unicycle robot

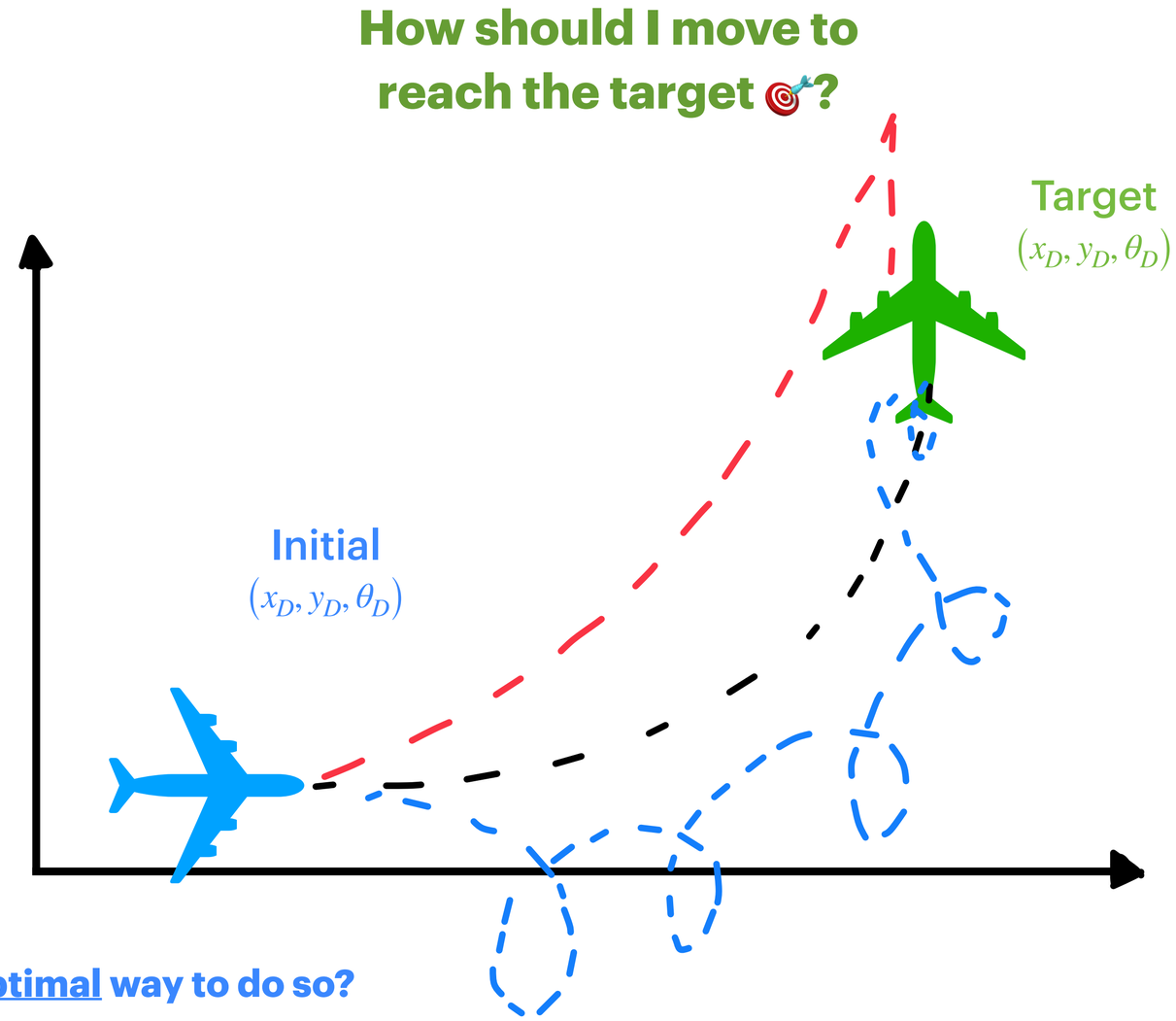
[Interactive lesson]

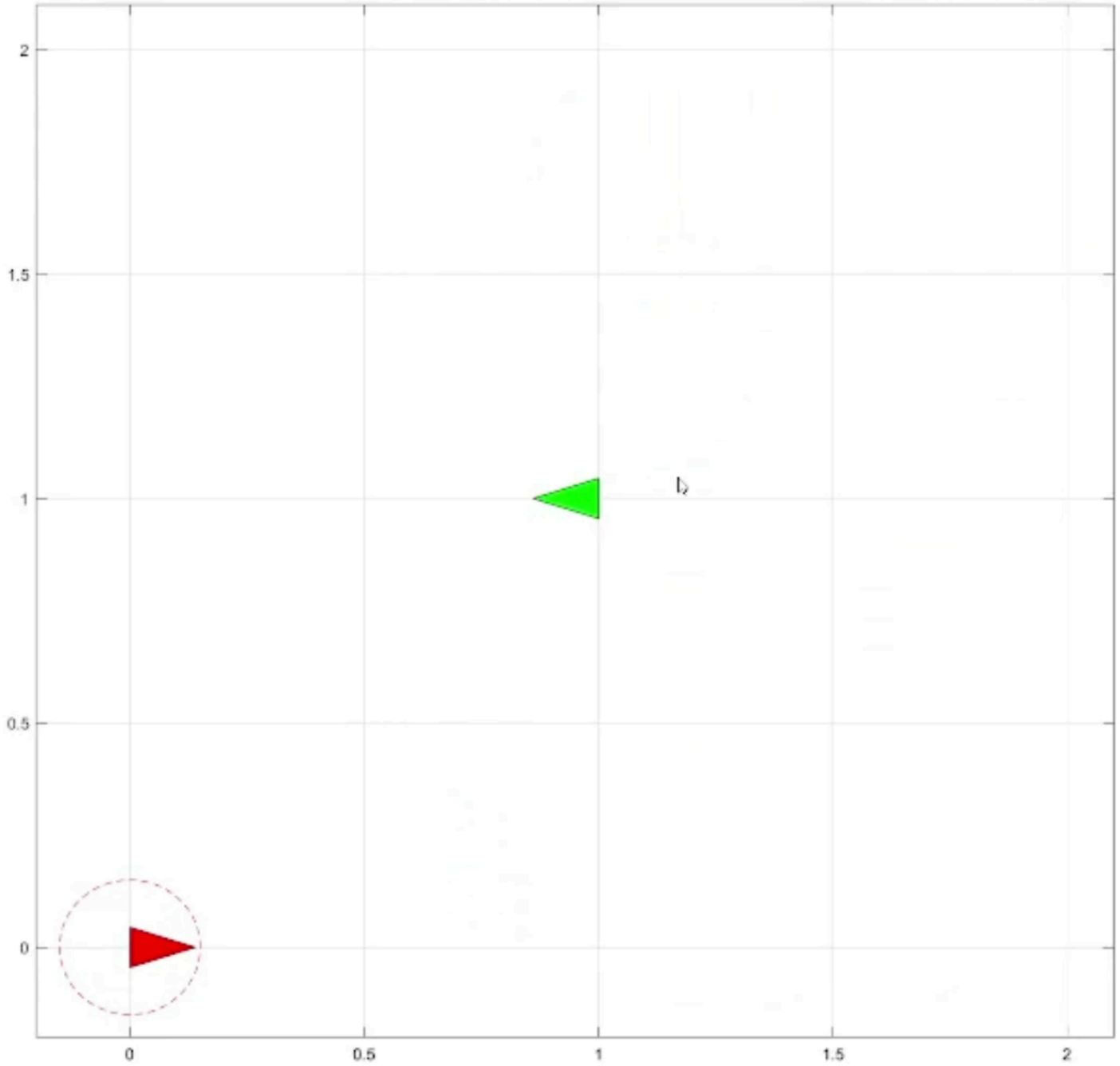
Alessandro Assirelli mat.231685

Task we are gonna cover

Reaching a state

Following a trajectory




2
1.5
1
0.5
0
0
0.5
1
1.5
2

Program of the lecture

1. Introduction to optimal control and how can we solve it
2. Super fast overview of CasADi
3. Trajectory optimization
4. MPC formulation

Introduction to OCP

- We are looking for an optimal control policy u(t) which optimally drives the system for the whole time horizon T
- x, u are trajectories, so infinitely dimensional
- Also the constraints are infinite

$$\min_{x,u} \int_0^T \varphi_R(x(t), u(t), t)dt + \varphi_F(x(T))$$

$$x(0) = x_0$$

Subject to $\dot{x} = f(x(t), u(t), t)$

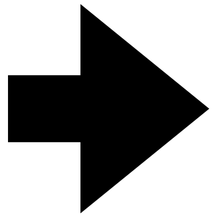
$$g(x(t), u(t), t) \leq 0$$

This is a discretization of the previous problem

$$\min_{x,u} \sum_{t=0}^{N-1} \varphi_R\left(x_t, u_t\right) + \varphi_F\left(x_N\right)$$

Subject to

$$x_0 = x_0$$
$$x_{t+1} = x_t + f(x_t, u_t)dt$$
$$g(x_t, u_t) \leq 0$$



General form of a NLP

$$\min_{w} \varphi(w)$$

Subject to

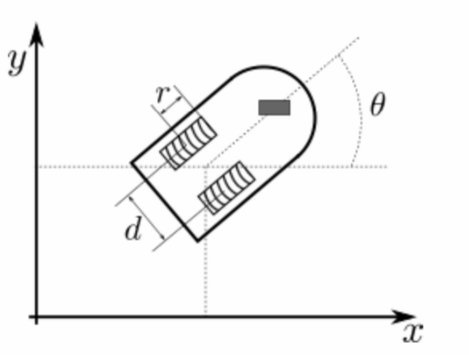
$$f(w) = 0$$
$$g(w) \leq 0$$

Now we have a NLP

We can use NLP off the shelves solver to find a <u>local optimum</u> 🙌🏽

What's about the unicycle

- We will focus only on the kinematics (no dynamics)
- We want to reach a reference state
- Also we want to keep the control low
- We may want to add velocity bounds



$$\begin{cases} \dot{x} = \dfrac{\omega_R + \omega_L}{2} r\cos(\theta) \\ \dot{y} = \dfrac{\omega_R + \omega_L}{2} r\sin(\theta) \\ \dot{\theta} = \dfrac{\omega_R - \omega_L}{d} r \end{cases}$$

$$\min_{z,u} \sum_{t=0}^{N-1} \| z_t - z_D \|^2 + \| u_t \|^2$$

Subject to

$$z_0 = z_0$$

$$z_{t+1} = z_t + f(z_t, u_t)dt$$

$$l_B \le u_t \le u_B$$

weights can be put inside or outside the NORM

You can find all the info here: https://web.casadi.org

Basically it's a library for automatic differentiation and it's very focused on optimal control. By installing it you are also installing some solvers. In particular Ipopt (ai pi opt for Italian speakers) is a very powerful one.

You can find a brief description of it here: https://drops.dagstuhl.de/volltexte/2009/2089/pdf/09061.WaechterAndreas.Paper.2089.pdf

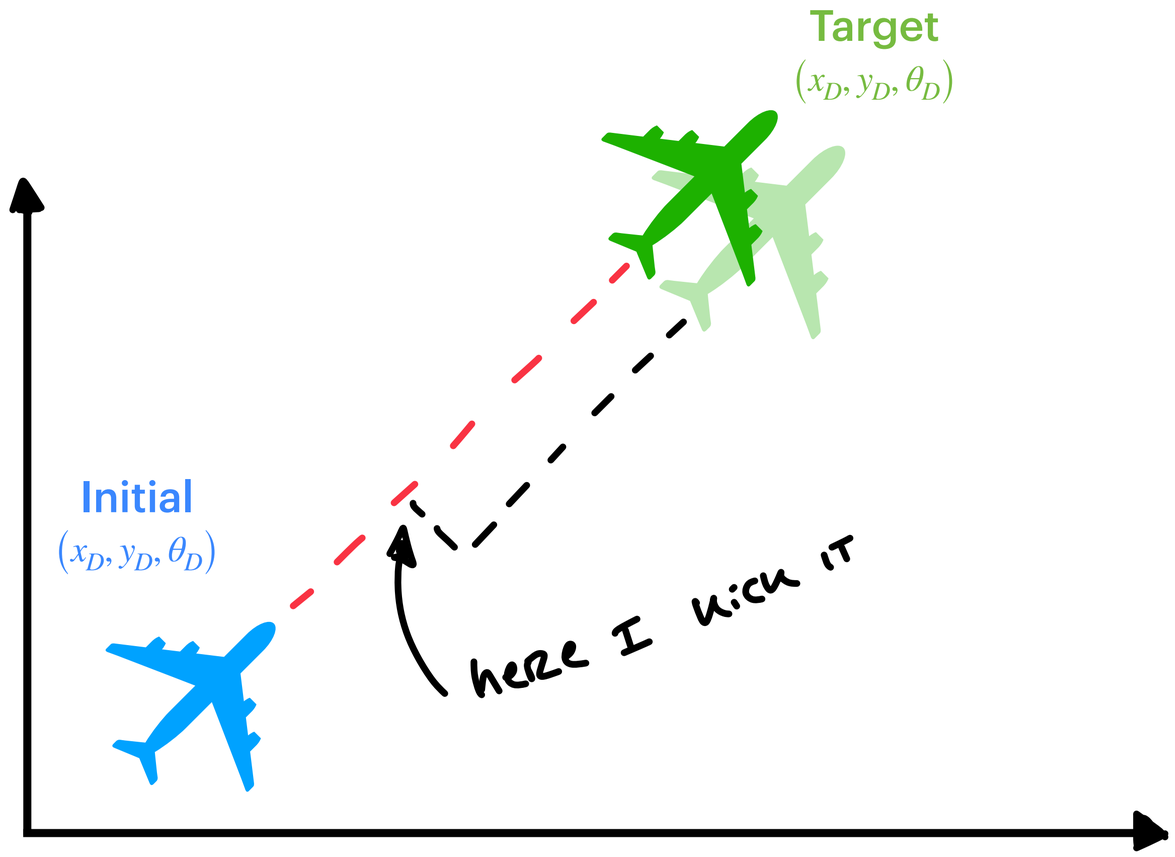
And a detailed one here: https://link.springer.com/article/10.1007/s10107-004-0559-y

How does it work is not the scope of this class and we will focus instead on how we can use it

Problem

If we apply directly u(t) we are gonna miss the target

We have to somehow close the loop



Solution

MPC is a way to use optimal control to control a robot.

It consist in solving an OCP from the initial state and apply the control only of the first time step while neglecting the others.

Then, another OCP is solved and it keeps running using only the first solution