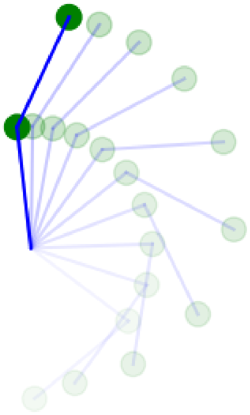
Underactuated Robots
Lecture 6: Examples

Nicola Scianca

September 2024

outline

- cart pendulum
- pendubot
- planar UAV
- humanoid robot



implementation

- the first three examples are written in Python, using **casADi** to formulate the optimization problem and **ipopt** to solve it
- the Python code is available at this **repository**:
`https://github.com/DIAG-Robotics-Lab/underactuated`

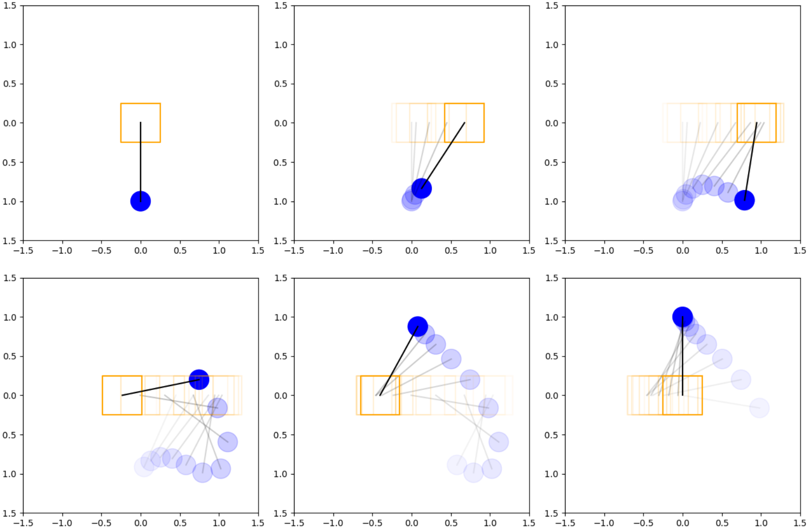
- the last example is written in C++, using **DART** (Dynamic Animation and Robotics Toolkit), and the optimization problem is solved with **HPIPM**

cart pendulum

- the **cart pendulum** consists of a pendulum swinging in the vertical plane attached to a cart moving on a horizontal track
- the state variables are
  - x position of the cart
  - θ angle of the pendulum with the vertical
  - $\dot{x}$ velocity of the cart
  - $\dot{\theta}$ angular velocity of the pendulum
- the trajectory optimization problem is

$$
\begin{aligned}
& \min \sum_{i=0}^{N-1} u_i^2 \\
\text{s. t.} \quad & x_{i+1} = f(x_i, u_i) \quad \text{for} \quad i = 0, \ldots, N-1 \\
& x_0 = (0, \pi, 0, 0) \\
& x_N = (0, \pi, 0, 0)
\end{aligned}
$$

cart pendulum

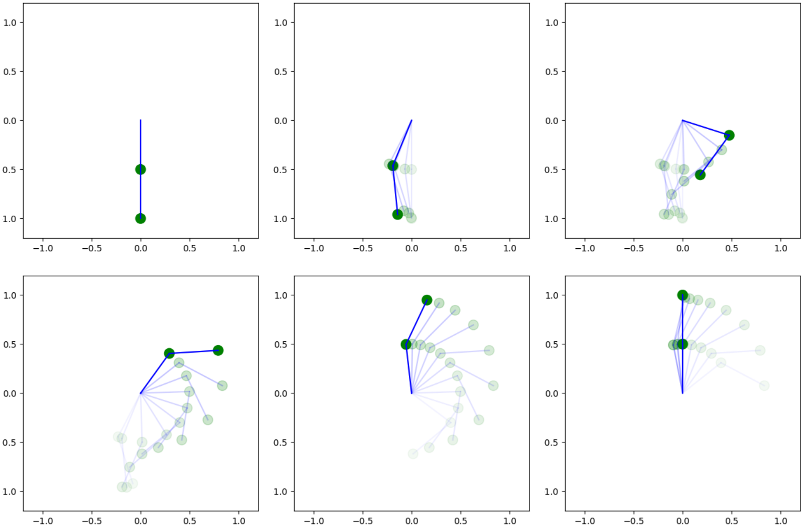


pendubot

- the **pendubot** is a two-link robot arm (double pendulum), where the first joint is actuated and the second is not
- the state variables are
  - θ_1 angle of the first link with the vertical
  - θ_2 angle of the second link relative to the first
  - $\dot{\theta}_1$ angular velocity of the first link
  - $\dot{\theta}_2$ angular velocity of the second link
- the trajectory optimization problem is

$$
\begin{aligned}
&\min \sum_{i=0}^{N-1} u_i^2 \\
\text{s. t.} \quad & x_{i+1} = f(x_i, u_i) \quad \text{for} \quad i = 0, \ldots, N-1 \\
& x_0 = (\pi, 0, 0, 0) \\
& x_N = (0, 0, 0, 0)
\end{aligned}
$$

pendubot

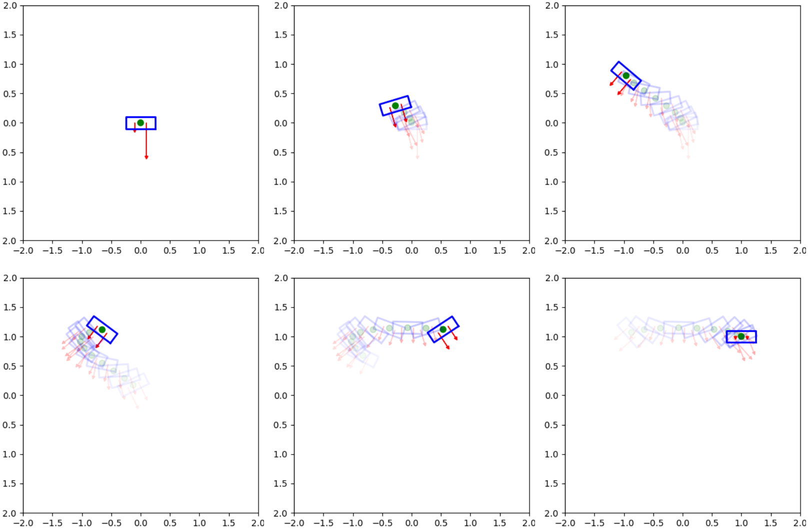


planar UAV

- the **planar UAV** consists of an aerial vehicle in the vertical plane, with two thrust forces applied at different points
- the state variables are
  - x horizontal position
  - z vertical position
  - θ pitch angle of the UAV
  - $\dot{x}$, $\dot{z}$, $\dot{\theta}$ velocities of the corresponding states
- the trajectory optimization problem is

$$
\begin{aligned}
\min \sum_{i=0}^{N-1} u_i^T u_i & \\
\text{s. t.} \quad & x_{i+1} = f(x_i, u_i) \quad \text{for} \quad i = 0, \ldots, N-1 \\
& x_0 = (0, 0, 0, 0, 0, 0) \\
& (x_{N/2}^x, x_{N/2}^y) = (-1, 1) \\
& x_N = (0, 0, 0, 0, 0, 0)
\end{aligned}
$$

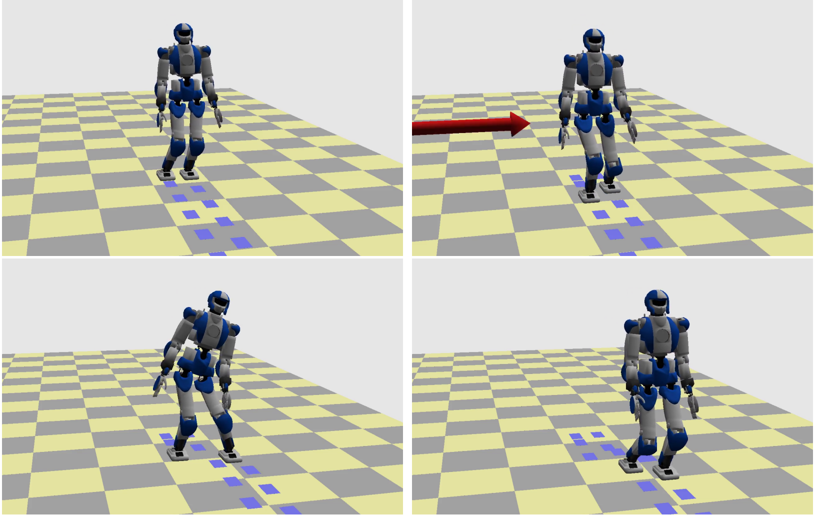
planar UAV



humanoid robot

- in this example, control of a humanoid robot is achieved via the interaction of different modules
- a **footstep planner** determines the position (and possibly the orientation and timing) of the footstep sequence
- an **MPC** generates the CoM and ZMP trajectory, according to a **simplified model** (LIP dynamics)
- a **whole-body controller** generates joint commands (here joint accelerations)
- a push is applied to test for robustness

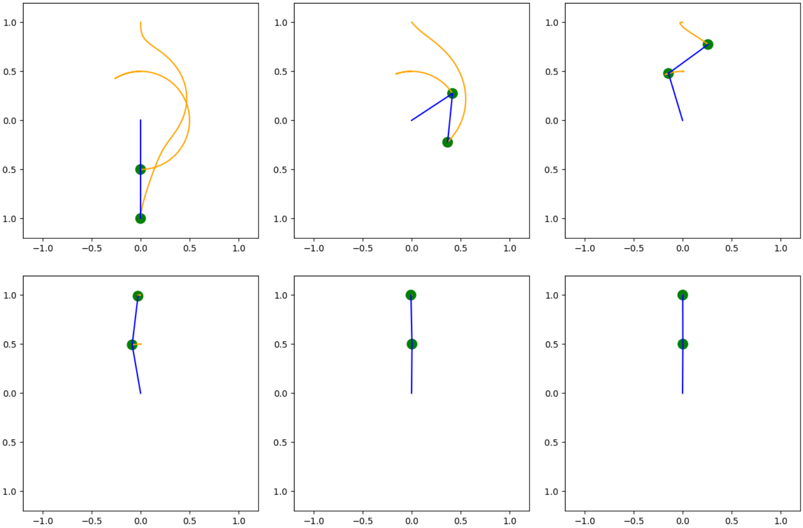
humanoid robot



pendubot MPC

- the next two examples show MPC control of a **pendubot**, with and without warmstarting the optimization to the previous solution
- if we warmstart the optimization to the previous solution evreything goes smooth and the swing-up task is correctly achieved
- if we don't do that, at some point the solution jumps to a different **local minimum**, and the pendulum gets stuck in a loop

pendubot MPC: with warmstart



pendubot MPC: without warmstart